CIS-83-01

Computer-Controlled Freeway Surveillance & Control System on the Queen Elizbeth Way

J. Tsai
Project Research Engineer
Control and Information Systems

Published by:
The Transportation Technology and Energy Branch
Ontario Ministry of Transportation and Communications
Hon. James W. Snow, Minister
H.F. Gilbert, Deputy Minister

Published without prejudice
as to the application of the findings.
Crown copyright reserved; however, this
document may be reproduced for non-commercial
purposes with attribution to the Ministry.

For additional copies write:
The Editor
Research and Development Branch
Ministry of Transportation and Communications
1201 Wilson Avenue
Downsview, Ontario
M3M 1J8

April 1983

TABLE OF CONTENTS

PAGE

PREFACE vii
1/ INTRODUCTION TO FREEWAY SURVEILLANCE AND CONTROL SYSTEM (FSCS) 1
1.1/ FSCS System Definition and Objective. 1
1.2/ Computerized QEW FSCS System Scope. 1
1.3/ QEW Freeway Corridor Traffic Management Capability. 3

2/ QEW FSCS HARDWARE DESCRIPTION 4
2.1/ Hardware System Configuration 4
2.2/ NOVA 4/X Traffic Control Computer 4
2.3/ System Keyboard Dasher Printer 6042 5
2.4/ Tektronix 4027 Colour Graphics Terminal 6
2.5/ Tektronix 4632 Hardcopy Unit. 7
2.6/ Disk Unit 7
2.7/ Magnetic Tape Unit. 8
2.8/ Universal Line Multiplexer (ULM) Unit 8
2.9/ Line Printer. 9
2.10/ Computer-to-Computer Data Link Unit. 9
2.11/ HP 21MX Communication Computer 10
2.12/ Flexible Disk Subsystem. 10
2.13/ Decitek Paper Tape Reader. 10
2.14/ ASR-33 TTY 11
2.15/ Head-end Master Data Modem 11
2.16/ Two-way Data Channels in a Coaxial Cable Network 12
2.17/ Field Data Modem Unit. 12
2.18/ Field Data Modem Input/Output Interface. 14
2.19/ TDM Coaxial Cable Data Communication Scheme. 15
2.20/ Coaxial Cable Data Communication Range 16
2.21/ Loop Detector Unit and Interface 16
2.22/ Ramp Controller Unit and Interface 17
2.23/ Changeable Message Sign Controller Unit and Interface. 19

3/ QEW FSCS REAL-TIME SOFTWARE DESCRIPTION 21

3.1/ System Operation Overview. 21

3.2/ FSCS Real-Time Task Analysis 22

3.3/ Mapped Real-Time Disk Operating System 23

3.4/ FSCS Real-Time Application Software. 24

3.5/ System Task Scheduling Program 25

3.6/ Communication Program. 26

3.7/ Traffic Surveillance and Control Program 27

    3.7.1/ Traffic Data Acquisition Function 30

    3.7.2/ Station Incident Detection Function 33

    3.7.3/ Lane Incident Detection Function. 34

    3.7.4/ Ramp Control Function 38

    3.7.5/ Field Hardware Monitoring Function. 42

3.8/ Traffic Display Control Task Program 44

3.9/ Corridor Control and Surveillance Task Program 48

3.10/ System Operator Control Command Function. 48

3.11/ System I/O Channel Usage Assignment 49

3.12/ System Parameter Definition 49

3.13/ QEW FSCS Program File Organization. 49

3.14/ QEW FSCS Data File Organization 49

4/ QEW FSCS TRAFFIC CONTROL COMMAND FUNCTION DESCRIPTION 51

4.1/ RC Second-level Command Group. 52

4.2/ QO Second-level Command Group. 55

4.3/ CMS Second-level Command Group 58

4.4/ G Second-level Command Group 62

4.5/ I Second-level Command Group 63

4.6/ H Second-level Command Group 67

4.7/ JCW Second-level Command Group 71

4.8/ X Second-level Command Group 76

4.9/ System Handshake Dialogue 76

4.10/ System Performance Evaluation Command 77

5/ QEW FSCS OPERATION PROCEDURE 78

5.1/ Bringing Up Nova 4/X Traffic Control Computer System 78

5.2/ Taking Down Nova 4/X Traffic Control Computer System 78

5.3/ Loading BDCSP Software on HP 21MX Computer. 79
5.4/ Bringing Up BDCSP Software on HP 21MX Computer. 79
5.5/ Bringing Up Diagnostic Software on HP 21MX Computer 80
5.6/ Loading QEW Software 80

6/ FINE TUNING OF QEW FSCS SOFTWARE 81
6.1/ Adjustment of Ramp Control System Parameters. 81
6.2/ Adjustment of Ramp Queue Override Control Parameters. 82
6.3/ Adjustment of Incident Detection System Parameters. 82
6.4/ Adjustment of Detection Hardware Monitoring Parameters. 84
6.5/ Generation of Changeable Message Sign Messages. 84
6.6/ Explanation of Colour Graphics CRT Traffic Status Display. 84
6.7/ Interface with Arterial Route Permanent Counting Stations . . . 85

7/ REFERENCES 86

APPENDICES 119
1/ The QEW Freeway System Scope 119
2/ The QEW Hardware System Scope. 120
3/ Conceptual Control System Configuration. 121
4/ The Control Computer Hardware System Block Diagram 122
5/ NOVA 4/X Control Computer CPU Chassis Configuration. 123
6/ HP 21MX Communication Computer CPU I/O Slot Assignment 124
7/ QEW Broadband Coaxial Cable Network Layout 125
8/ New QEW Control Centre and OPP Radio Room Coaxial Cable Layout. 126
9/ Coaxial Cable Network Tap Station Carrier Level. 127
10/ Broadband Amplifier Input and Output Level 128
11/ The Connection between Data Modem and the Coaxial Cable Tap Station. 130
12/ Modem Address and Traffic Device Assignment. 131
13/ Cable Delay Modification 134
14/ CR710 Ramp Controller Ramp Queue Override Function 141
15/ CR710 Interface Functional Assignment. 143
16/ Sample of Sign Display and Format. 147
17/ CMS Sign Character Symbol and ASCII Code 148

18/ CMS Controller Input Message Format and the CMS Controller Response. 149
19/ CMS Controller and Data Modem Interface Connector. 154
20/ QEW FSCS Real-Time Software Functional Block Diagram 155
21/ QEW Real-Time Application Software Structure and Program Size 156
22/ Real-Time Multi-Task Program Scheduling Sequence 158
23/ "DETECT" Traffic Data Buffer Format. 159
24/ "COMDOP" Ramp Control Command Buffer Format. 173
25/ Computerized Ramp Metering Rate Selection Logic. 174
26/ Ramp Queue Detection Logic and Ramp Queue Override Feature. 184
27/ DATA30 File Logical Record Format. 190
28/ DATA5M File Logical Record Format. 206
29/ QEWDYN File Logical Record Format. 216
30/ INCLOG File Logical Record Format. 227
31/ ALMREC File Logical Record Format. 228
32/ XMTTAB File Logical Record Format. 231
33/ RESDEV File Logical Record Format. 233
34/ WEATHER File Logical Record Format 235
35/ First Physical Record Format of Data Files 237
36/ QEW FSCS Real-Time Software Source Program Listing 250

LIST OF TABLES

PAGE

1/ Maximum Controllable Cable Length. 87
2/ Loop Centre to Loop Centre Distance. 88
3/ CR 710 Ramp Controller Interface Assignment. 89
4/ Type 170 -- Remote Ramp Control Command Definition 90
5/ Operator Command Function and Overlay Subroutines. 91
6/ Traffic Control Computer I/Q Channel Usage 92
7/ Control Software Parameters. 94
8/ QEW Data File Structure. 108
9/ Ramp Metering Supports File Structure. 109

LIST OF FIGURES

1/ First-Level Command Structure. 110
2/ Second-Level Command Structure Under QEW 111
3/ Third-Level Command Structure Under "RC" 112
4/ Third-Level Command Structure Under "QO" 113
5/ Third-Level Command Structure Under "CMS". 114
6/ Third-Level Command Structure Under "G". 115
7/ Third-Level Command Structure Under "INF". 116
8/ Third-Level Command Structure Under "H". 117
9/ Third-Level Command Structure Under "JCW". 118

PREFACE

The main purpose of this report is to describe the QEW control software. However, since the function of the control software is to respond to the real-time highway environment, and to activate and monitor the control hardware, the interfaces among hardware, software and highway environment are also described.

The manuscript of this report was completed in the summer of 1981. It reflected the information available at that time.

1/ INTRODUCTION TO FREEWAY SURVEILLANCE AND CONTROL SYSTEM (FSCS)

1.1/ FSCS System Definition and Objective

The principal goal of a freeway surveillance and control system is to maintain an acceptable level of traffic service on the freeway. The advent of the computer-supervised freeway surveillance and control system adds a significant new dimension to the terms surveillance and control. Weinberg (1964) defined computer-supervised surveillance and control as:

> "A closed loop system .... that provides for surveillance of traffic operations which can be processed by a computational logic in a real-time computer, testing observed conditions against a set of decision rules, selecting commands in light of the results of the test, activating appropriate controls or communicating with drivers to improve traffic movement when necessary, and then reassessing the traffic behaviour to determine if further corrections are to be made."

Thus, it can be summarized that a computerized freeway surveillance and control system is one which provides the optimal real-time management of freeway corridor traffic, particularly in providing the quantitative data required to effectively manage traffic bottlenecks and capacity-reducing incidents.

1.2/ Computerized QEW FSCS System Scope

This section describes the expanded QEW FSCS system scope from the freeway, hardware, and software points of view.

Freeway: The freeway system includes Highway 403 (Ford Drive) interchange, Winston Churchill interchange, Southdown Road interchange, Mississauga Road interchange, Highway 10 interchange and Cawthra Road interchange. The controlled freeway section is approximately 16.64 km long. There are 22 mainline freeway speed sampling loop detector stations which are approximately 0.8 km apart. The function of the mainline detector station is to assess the level of service on the mainline freeway. Each interchange's on-ramp has queue, demand, and passage detectors installed to assess the on-ramp traffic during the morning rush hour period. Each interchange off-ramp has an exit detector installed to determine the off-ramp traffic. Each on-ramp has a ramp controller and a signal light system installed to control on-ramp traffic during the morning rush hour period. Appendix 1 shows the freeway system scope.

Hardware: The hardware system includes the traffic control computer subsystem, the traffic data communication subsystem, the ramp controller and signal light subsystem, the traffic detector subsystem and the changeable message sign subsystem. The traffic control computer subsystem includes a NOVA 4/X central processing unit with 128K words of memory, a 10-million byte disk unit, a 9-track magnetic tape unit, a Dasher control console, a Tektronix colour graphic display unit, a Tektronix hardcopy unit, and a front-end traffic communication processor. The traffic data communication subsystem includes the front-end traffic communication processor with 32K byte memory, and a head-end master data modem, an ASR-33 TTY, twenty-five field data modems and two-way data channels provided by a two-way coaxial cable system. The ramp controller and signal light subsystem includes five CR710 ramp controllers, three type 170 ramp controllers, and the associated signal light equipment. The traffic detector subsystem includes 180 loop detectors. Appendix 2 shows the hardware system scope.

Software: The software system includes a mapped Real-Time Disk Operating System (MRDOS), traffic data acquisition subsystem, incident detection subsystem, ramp control subsystem, changeable message sign subsystem, freeway corridor status display subsystem, hardware monitoring subsystem, and system control command subsystem. The function of the QEW FSCS software is to effectively manage the preceding hardware system resources in responding to dynamic and changing freeway corridor traffic behaviour such that an acceptable level of service for the controlled freeway corridor can be obtained.

The QEW FSCS software ("QEW") is running under MRDOS in a real-time multi-task environment. The software is designed in real-time FORTRAN 4 and NOVA assembly language.

1.3/ QEW Freeway Corridor Traffic Management Capability

Since the implementation of the QEW FSCS, there have been considerable changes in the characteristics and capacity of the QEW corridor network. Meanwhile, the City of Mississauga is planning to implement a street traffic signal control system. The combination of these events has created an opportunity to implement a QEW corridor control system. Such a system will improve overall utilization of the corridor highway network facilities including freeway and arterial routes.

As a result, the system hardware and software configuration is flexible enough for future expansion to satisfy the requirement of the corridor control system. For example, two permanent counting stations near the Credit River Bridge -- one on Lakeshore Road (Highway 2), and another on Dundas Street (Highway 5) -- have been installed and connected to the QEW FSCS freeway management centre through the public telephone line network for traffic surveillance purposes. The interconnection between QEW FSCS and the Mississauga street traffic control system is also under study and planning.

2/ QEW FSCS HARDWARE DESCRIPTION

2.1/ Hardware System Configuration

The expanded QEW FSCS project has 180 loop detectors, 8 ramp controllers, and 1 changeable message sign installed along 16.64 km of the eastbound QEW freeway section between Highway 403 interchange and Cawthra Road interchange. These traffic sensing and control devices are connected to 25 field data modems located inside the 25 field cabinets. Appendix 3 shows the control system configuration diagram for the expanded QEW FSCS project.

These 25 field data modems communicate with a head-end master data modem located in the traffic control centre through a duplex data channel provided by a two-way coaxial cable network.

In the traffic control centre, the head-end master data modem is connected to a communication processor (HP 21MX) which serves as a front-end preprocessor to control the time division multiplexing traffic data network and communicate with the NOVA 4/X traffic control computer. An ASR33 TTY with paper tape punch and reader facility is also connected to the communication processor for the purpose of monitoring system hardware status. A Decitek high-speed paper tape reader and a flexible disk subsystem are used for loading communication processor system software. The traffic control computer communicates with the communication processor through a duplex DMA data channel. The traffic control computer also controls a system keyboard (Dasher terminal), a colour graphics display unit (Tektronix 4027) and an attached hardcopy unit, a disk storage unit (6045 disk cartridge unit), a magnetic tape unit (6027 800 BPI tape drive), and a four-line asynchronous multiplexer. Appendix 4 shows the control computer hardware system block diagram.

2.2/ NOVA 4/X Traffic Control Computer

The Traffic Control Computer (TCC) system's Central Processing Unit (CPU) is the Data General NOVA 4/X which is a modular, reliable, general purpose mini-computer. The NOVA 4/X uses 16-bit, single-word multi-function instructions. The central processor uses a single 15" board to implement all CPU functions with on-board asynchronous interface, real-time clock, auto program load. It has four-way interleaved, 256KB MOS memory with 400 nanosecond cycle time. An instruction prefetch processor looks ahead of currently executing operations to store further instructions in a high-speed first-in, first-out (FIFO) buffer. It has a memory management and protection unit which allows protected two-ground operation, protection for I/O devices, and both write and validity protection for main memory. A 16-slot chassis provides maximum configurability. A virtual console further enhances reliability by eliminating almost all mechanical switches. At power-up time, a diagnostic program tests the central processor, memory, and the prefetch processor. The I/O system has a 16-bit word length, low-speed programmed data transfer, a high-speed direct memory access data channel, a 16-level programmed priority interrupt, a 64-device addressing capability, and automatic interrupt source identification. It also has a power failure monitor and hardware multiply/ divide capability. Appendix 5 shows the 16-slot chassis CPU configuration. Because most options are on the CPU board, and up to 256KB is available on a single memory board, the system has as many as 14 slots available for I/O controllers. The TCC is responsible for overall control of the system. It is programmed to analyze traffic data, make traffic control decisions, and to monitor hardware status. The TCC is connected to a satellite computer (HP 21MX) by a high-speed parallel data link. This link includes a specially adapted interface card in the NOVA 4/X. The TCC is responsible for controlling the computer-computer data link according to a specified protocol.

2.3/ System Keyboard Dasher Printer 6042

The system control keyboard Dasher printer is a fast, quiet unit for data input and output application. The 6042 has a standard 96-character, upper- and lower-case typewriter keyboard and provides both local and remote operation. It is interface compatible with teletype and other ASCII input devices. A 40-character buffer and a 60 cps catch-up printing capability give the 6042 printers true 30 cps throughput capability. When operating in the 30 cps mode, characters flow through the buffer and are printed at 30 cps, leaving the buffer empty. During carriage returns, the print head is disabled and characters accumulate in the FIFO buffer. When the carriage reaches full left position, the print head begins printing characters at 60 cps and continues in this catch-up mode until the buffer is empty. It then returns to 30 cps operation. The 6042 also has a user-selectable view mode. If no data are received for two seconds, the printing head automatically moves six spaces to the right, giving the user a clear view of the previously printed text. If the print head is within six spaces of its far right position, it automatically moves to the left. The print head moves back to its original position and begins printing when a character is received.

2.4/ Tektronix 4027 Colour Graphics Terminal

The Tektronix 4027 interactive graphics terminal has a 33 cm diagonal CRT display with 19.1 cm x 25.5 cm usable display area. It uses standard 525 master line scan with 480 lines displayed at 30 Hz scan rate. The display refresh rate is 30 times per second for the dot, 30 times per second for the frame, and 60 times per second for the field. The 4027 has a full ASCII keyboard with typewriter keys, terminal function control, programmable function keys, numeric pad, cursor control, scrolling, and crosshair cursor function. It has 80 characters per line and 34 lines per display with 2720 total characters per display. The character size is 7 x 9 in an 8 x 14 dot matrix. Each graphic cell (character position) is also an 8 x 14 dot matrix. The column of dots are numbered from left to right across the graphic region, starting with 0 for the left most column, and from bottom to top, starting with 0 for the bottom row. This establishes a coordinate system in the graphic region. For each dot in this region there is a pair of numbers: its X- and Y- coordinates. The potential number of dots is 640 horizontally and 476 vertically. Graphic input capability consists of a graphic crosshair cursor controlled by graphic cursor keys. In addition to reporting the coordinates back to the host computer, the 4027 also reports the colour of the designated coordinate. Since the location of the user-controlled crosshair can be read by the host computer along with the value of any key that is depressed, an interactive graphic dialogue can be established to edit or modify displayed data. Colours are selected from a 64 colour palette with up to eight colours displayed simultaneously on the screen. Features of the 4027 include colouring vectors, characters, symbols, and filling polygons. The firmware enables another colour to border the polygon.

The 4027's virtual bit map technique allows scrolling of colour alphanumerics and graphics on the screen. The 4027 has 32K bytes display memory to hold and display text, forms, and graphics and has 192K bytes of graphics memory to cover 4088 graphic cells. The device has an RS232C interface to the NOVA 4/X host computer with selectable data transmission speed up to 9600 baud. The standard video signal output allows connection of the 4027 to external video displays for group viewing and presentations.

2.5/ Tektronic 4632 Hardcopy Unit

The 4632 video hardcopy unit is used to make grayscale presentation of the 4027 colour display. A paper copy is produced when the front panel "copy" push button is pressed, or a remote signal (from 4027) is applied. The 4632's dry-process development system uses 3M Brand Type 777 Dry-Silver paper for the high image quality required by complex graphics and alphanumerics. It produces dry 21.6 x 27.9 cm ($8^1/_2$ x 11 inch) copy of 4027 display in grayscale and allows clean, conventional paper handling -- one can write on it in pen or pencil and erase pencil marks without damage. The 4632 produces the first copy in about 18 s, and subsequent copies of the same display in about 8 s.

2.6/ Disk Unit

The model 6045 cartridge disk subsystem includes one disk drive, power supply, a controller, cartridges, and cable sets. Each disk drive contains a single spindle for one two-surface fixed 5-megabyte disk cartridge and one two-surface removeable 5-megabyte disk cartridge. Each surface has a corresponding read/write head. The head positioning system provides fast radial positioning of the four heads to any one of the 408 cylinders. The average positioning time is 38 ms.

Track-to-track head positioning takes a maximum of 8 ms. Maximum full stroke positioning time is 70 ms. The disk drive operates at 2400 rpm with an average latency time of 12.5 ms and transfer data at 312 500 bytes per second. A special circuit monitors the temperature gradient between the cartridge and the driver after a new medium is loaded, and provides for a safe temperature differential before operation is begun.

This increases reliability and helps ensure data integrity. The disk drive requires 30 s to reach operating speed and 25 s to stop. The frequency requirement is 60 Hz ± 1 Hz and the power requirements are 120 VAC, 60 Hz and 4.9 A. The operating temperature range is 16°C to 32°C and the temperature gradient is 8.3°C/h maximum. The operating relative humidity range is 20% to 80% non-condensing. There are four removable disk cartridges (IBM 2316-type cartridge) in the system.

2.7/ Magnetic Tape Unit

The model 6027 magnetic tape subsystem includes control for 9-track 75 ips transport which is 800 bpi operation. The transport features uniform-tension vacuum columns designed to protect tape-stored data from damage. The 6027 uses 10 1/2" reels with 2400' of 1/2" tape. At 800 bpi, the 6027 9-track transport can store over 20 million bytes. Data transfer rate is 60 000 bytes per second via direct memory access channel. Internal microprocessor-based controller buffering of up to 16 bytes provides device latency of 250 μs. The start and stop time is 5 ms and the tape rewind speed is 200 ips nominal. The operating temperature range is 15.5°C to 32.2°C and the operating relative humidity range is 20% to 80% non-condensing. The power requirements are 120 VAC, 60 Hz and 5.5 A.

2.8/ Universal Line Multiplexer (ULM) Unit

The model 4241 ULM-5 unit provides four asynchronous lines in full-duplex operation and is a 15-inch square printed circuit board that occupies a single slot in the NOVA 4/X chassis. The ULM converts serial communications line data to parallel character transfer across the I/O bus under program control. It provides full-character buffering on all lines for transmitting or receiving data, and full modem control including auto answer for each line interface. All line interfaces are under independent program control. Each of the ULM's four asynchronous line interfaces can be programmed for parity type, code level, number of stop bits and line speed. When enabled, character parity is calculated on transmission and detected on reception. Each asynchronous interface can operate at selectable speeds up to 9600 bps. The line interface is EIA RS232C. The modem control covers Bell 103, 113, 202, 121 or equivalent.

The modem control signals are carrier detect, data set ready, ring indicator, request to send, data terminal ready, clear to send. The analog test loopback feature lets users test internal data paths, including line drivers and receivers, under program control. Individual receivers on the ULM can detect parity errors, overrun conditions and framing errors (missing stop bits). The ULM diagnostic mode facilitates separation of any hardware or software problems in the system from external data line conditions.

2.9/ Line Printer

The model 4034G 300 LPM line printer includes 4014 I/O interface and 4193 line printer controller. The 4034G has a 64-character ASCII set and operates at 300 lines per minute. It has 136 print positions per line and prints 10 characters per inch. A 136-character line buffer receives and stores the data to be printed. Each line is printed in two impressions, one for even-numbered character columns, one for odd-numbered character columns. The characters are etched on the surface of a continuously rotating metal drum. Printing occurs when a print hammer assembly strikes the drum through paper and inked ribbon. The hammer assembly contains 68 hammers, and moves laterally to print the odd and even character columns. The hammer has a life of over 500 million operations. The printer uses standard 11" fanfold, edge-punched, continuous form paper. An original and up to five copies can be produced.

2.10/ Computer-to-Computer Data Link Unit

The NOVA 4/X computer to HP 21MX computer data link unit is supplied by the data communication system supplier (IS/3M, Inc.). It interconnects the two CPUs in a master-slave relationship (the NOVA 4/X is master, the HP 21MX is slave). The NOVA 4/X data link interface is a Data General general-purpose interface board with the 16-bit input and 16-bit output data register option, and data channel (DMA) connection option. The HP 21MX data link interface utilizes the HP DMA facility and the program interrupt facility to transfer data. The two interfaces are connected by a duplex channel cable no more than 20 feet long.

2.11/ HP 21MX Communication Computer

The broadband data communication system's Central Processing Unit is a Hewlett-Packard HP 21MX computer which is a powerful microprogrammable processor with 128 standard instructions. The central processor has four general purpose registers, two of which may be used as index registers. It has a 16K word semiconductor memory expandable to 64K words. The memory cycle time is 650 ns. The initial binary loader is ROM resident and callable by a push-button switch on the operator panel. The central processor has up to two hours of memory sustaining power with an optional power fail recovery system. It has a Dual-Channel Port Controller with bidirection input/output transfer, which provides direct memory access, program assignable to any I/O channel. There are nine I/O channels in the mainframe with a multi-level vectored priority interrupt facility. A time base generator is available for real-time data communication system control purposes. The HP 21MX contains interfaces for computer-computer channel, teletype, paper tape reader, head-end modem and flexible disk drive. Appendix 6 shows the HP 21MX I/O slot assignment.

2.12/ Flexible Disk Subsystem

The model 12732A flexible disk subsystem consists of an HP 9885M flexible disk drive, and an HP 12735A computer interface. The subsystem is designed for use with the HP 21MX computer. The purpose of this device is to load the QEW data communication system program or communication network diagnostic program at very high speed from the flexible disk. The flexible disk drive is a random-access, mass-storage device with a capacity of up to 514 300 bytes on one removeable diskette. The flexible disk drive is self-contained and incorporates a self-test feature. The disk drive has its own internal power supply.

2.13/ Decitek Paper Tape Reader

The purpose of this device is to load a system program at high speed when processor power is off for more than two hours. The paper tape reader speed is 300 cps (characters per second).

2.14/ ASR-33 TTY

This device monitors communcation system hardware status and allows stand-alone mode changeable message sign control. The printer speed is 10 characters per second. It also has a paper tape punch and reader facility. The paper tape reader speed is 10 cps. It can also serve as a backup unit for the high-speed tape reader.

2.15/ Head-end Master Data Modem

The head-end master data modem is an ISI 310 modem which handles all RF transmitting and receiving, logic functions, and serial/parallel conversions required at the head-end of a cable network. It is connected to the communication processor HP 21MX through an interrupt-driven 16-bit duplex interface. The ISI 310 head-end modem performs logic functions which include appending data received on the upstream channel with the proper modem address and interrupting the processor to which it is connected. It has an RF transmit and receive card, an RF-to-digital conversion card, and a control logic card. Only one head-end master data modem is required to control the coaxial cable network. An additional unit is available as a spare. The 310 head-end is configured to run on a sub-split two-way broadband coaxial cable system with transmitting channel centre frequency at 28.8 MHz and receiving channel centre frequency at 238.5 MHz. It is designed to handle the transmission delays caused by a long coaxial cable. With 48 000 bits per second data rate, the maximum allowable coaxial cable length from the head-end to the field station is approximately 32 km. It has the following throughput characteristics:

- 48 000 bits per second data rate
- 13 bits per word in transmitting channel
- 3692 words (time-slots) per second (48 000/13)
- 148 devices for 1/25 second polling time per device address (3692/25)
- 144 usable device addresses, since the last four addresses must be used for head-end reset.

2.16/ Two-Way Data Channels in a Coaxial Cable Network

The two-way upstream and downstream data channels to and from the traffic control centre are provided by a sub-split two-way broadband coaxial cable network. The upstream channel bandwidth is within the 50 to 250 MHz range, and the downstream channel bandwidth is within the 5 to 30 MHz range. The current downstream data channel carrier frequency is centred at 28.8 MHz. The upstream data channel carrier frequency is centred at 238.5 MHz. After 1980 system expansion, the current coaxial cable network consists of two main trunk lines. The east trunk line extends from trailer control centre (located at the Highway 10 interchange) to the Cawthra Road interchange; the west trunk line extends from the control centre to the Ford Drive interchange. Appendix 7 shows the QEW broadband coaxial cable network layout. Appendix 8 shows the new trailer traffic control centre and OPP station radio room coaxial cable layout. Appendix 9 shows the coaxial cable tap station carrier level. Appendix 10 shows the system input and output levels at the broadband amplifiers. Appendix 11 shows the connection between the data modem and the tap station.

2.17/ Field Data Modem Unit

The system contains 1-address and 4-address modems. The 4-address modems contain the following cards:

- RF (radio frequency modulator and demodulator with transmitting frequency of 238.5 MHz and receiving frequency of 28.8 MHz)
- serial
- control
- timer
- 32-bit input

The 1-address parallel I/O modems contain the following cards:

- RF
- serial
- control
- timer (in 1980 system expansion model only)
- 32-bit input (in 1980 system expansion model only)

- 8-bit input (in the original system model only)

The sign control modem, a 1-address modem with RS232C I/O contains:
- RF
- serial
- control
- timer
- BRC (baud rate converter)

The RF, serial, and control cards combine to communicate with the head-end on coaxial cable and handle parallel 8-bit I/O. The 32-bit input cards buffer up to 32 bits of input connected to a modem. Appendix 12 shows the assignment of modem addresses to traffic devices. Modems numbered 1 to 11 are in the original system and modems numbered 12 to 25 are in the expansion system.

On the modems in the original system, output to the parallel I/O devices goes to the internal latch, using control code 3. (A jumper on the control card determines whether the internal latch is loaded by control code 3 or 6.) When a 3 s timeout occurs, the low-order bit (bit 0) of each output byte is changed from a 1 to a 0. The 3 s timeout is reset each time a control code 3 is received by the field modem.

On the modems in the system expansion, output to the parallel I/O devices goes to the timer card using control code 3, and the internal latch is jumpered to control code 6. The timer card asserts a specified output signal from the modem if a 3 s timeout occurs. In fact, the timeout sets the entire output byte to 0.

On the modems with RS232C I/O, the timeout turns off the carrier detect signal (pin eight of the 25-pin RS232C connector). The external devices (i.e., ramp controllers or sign controllers) use these timeout signals to take local control action.

On the changeable message sign control modem, a BRC card has a 256-byte output buffer for storing data sent from the head-end at a higher baud rate. The data are retransmitted to a sign controller at a lower baud rate according to RS232C specifications. Baud rate (switch-selectable on the BRC cards) is set to 300 for the changeable message sign controller. Data are sent from the head-end to the BRC cards using control code 0.

2.18/ Field Data Modem Input/Output Interface

Input to Modems

Each new 840 modem has a 32-bit TTL input card, which handles up to 32 parallel input bits from devices with open collector outputs. Input levels are as follows:

Logic Level	Input Signal
0	Open circuit, which input card pulls up to +5 V
1	Ground, for which external device must sink approximately 20 mA

These modems are configured for internal strobing, so that they read the data on the input lines every time a head-end reset occurs (approximately 25 times per second).

Output from Modems

Each new 840 modem has a new timer card, which buffers output data from the computer, provides eight parallel output bits to a ramp controller, and notifies the controller if communication with the head-end is broken. Each timer card has a 3 s timer which is reset every time the Traffic Control Computer sends data to the modem. If the modem receives no data from the computer for 3 s the timer card sets all eight output bits to 0.

The ramp controller then enters local control. Each timer card provides open collector TTL outputs. Output levels are as follows:

Logic Level	Output Signal
0	Open circuit, which ramp controller pulls up to +12 V
1	Ground, capable of sinking approximately 40 mA from ramp controller

For system testing, the timer cards may be used to generate input to the modem via wrap-around connections. Each timer card has a DIP switch which selects the address modulus 4 of the output byte from the modem. For instance, if address 17 octal had a 1-address modem, then setting the timer cards switch for address 3 would cause the proper output byte to be sent out.

2.19/ TDM Coaxial Cable Data Communication Scheme

Downstream data are the data sent from the head-end along the coaxial cable to the field data modem. Upstream data are the data sent by each remote data modem back along the coaxial cable to the head-end.

Downstream data communication is accomplished on a single time-shared channel with a modem addressing scheme. To send data to a traffic device connected to a particular modem address, the head-end first selects the unique address in the data modem, sends the data and then deselects the modem address. Each downstream data word consists of 1 start bit, 3 bits of control code followed by 8 bits of data and 1 stop bit. The three control bits generate eight combination control words.

Upstream data communication is accomplished by a Time Division Multiplexing (TDM) System. The same modem addresses used in the downstream data channel are used to define a sequence of time slots in the single upstream data channel. Therefore, the modem addresses "n" will have the opportunity to send upstream data during the Nth time slot. Although the time slot addresses are fixed in the hardware, the head-end modem uses control codes to ensure that the phase counters (which control the upstream data channel time slots in all the modems) are synchronous with one another by issuing a "Run Phase Counters from Zero", every 148 downstream data words.

Besides the normal data communication mode of operation, the data modems may be set into an enquiry state. In this state the upstream channel data are inhibited and all field data modems send their device address in their respective time slot. This is a positive indicator to the head-end that the greater part of the modem circuitry is operating correctly.

2.20/ Coaxial Cable Data Communication Range

In the 1980 system expansion, the coaxial cable has been extended to longer than 12.8 km (8 miles) which is beyond the existing head-end modem capability for controlling the remote modem. As a result, the head-end modem was modified by IS/3M according to a cable delay modification design change which allows the head-end to control longer coaxial cable. Essentially, the cable delay is the time it takes for a signal, travelling at approximately the speed of light, to go from the head-end to a remote modem and back. Maximum controllable cable lengths before and after the modification are shown in Table 1.

Appendix 13 describes the details of the cable delay modification which may allow the head-end to control the coaxial cable up to 128 km in length when the system is operating at a 12 000 bits per second data rate.

2.21/ Loop Detector Unit and Interface

The original system loop detector unit is the Traffic Data Systems Inc. Loop Detector series LD328 which is designed to operate with a wide range of loop configurations. After initial manual tuning, tracking of environmental changes is automatic. The power supply is an integral part of the detector unit, with all controls mounted on the front panel . Two sensitivity levels, three presence modes, and four operating frequencies enable the detector unit to accommodate the usual configurations of loops and lead-ins.

The loop detector unit provides reliable detection by phase comparison means with an accuracy of better than 99% of all vehicles passing over the loop at speeds of 1 to 80 mph. The operation of the detector unit is not affected by changes in the inductance of the loop caused by environmental changes. The output of the (LD328 loop) detector unit is a fail-safe relay with isolated contacts or a solid state NPN open collector output. A matching circuitry interface box is supplied by ISI for each data modem to connect loop detector unit output to the data modem.

There are 77 loop detectors installed for the original QEW FSCS demonstration project. Table 2 shows the mainline speed sampling loop detector spacing. All the mainline loops have a size of approximately 6' x 6'.

In the 1980 system expansion, the number of mainline detector stations was increased to 22 and the total number of loop detectors was increased to 180.

2.22/ Ramp Controller Unit and Interface

After 1980 system expansion, there are two types of ramp metering controller, the TDS CR710, and the Type 170.

CR710 Ramp Controller
The TDS CR710 ramp controller used in the original QEW system is designed for metering traffic onto freeways from conventional access ramps. They are installed at the Highway 10 N/B, Highway 10 S/B, Mississauga N/B, Mississauga S/B, and Southdown Road entrance ramps.

The controller sequences through green, yellow, and red signal intervals, allowing motorists to enter the traffic flow at a selected rate. Demand and passage loops enable the controller to operate in either loop-actuated or pretimed modes, allowing vehicles on the ramp to enter on a demand basis or at a pretimed rate.

Traffic flow rate can be chosen by three different modes:

1/ Time-of-day clock choosing one of six preloaded cycle lengths. This mode can also be activated by a relay controlled hardware switch located in the control centre.

2/ Mainstream occupancy actuated, in which the CR710 chooses one of six preloaded cycle lengths based on mainstream occupancy.

3/ Remote computer mode, in which the CR710 operates as a slave controller executing the computer commands. The computer commands override the mode 1 or mode 2 operation. To relieve the potential entrance ramp traffic congestion, a ramp queue override feature was implemented under local control operation modes (i.e., modes 1 and 2).

Appendix 14 describes the CR710 ramp queue override function. Reference 1 describes the details of the CR710 ramp controller operation principle.

The data modems provide 8-bit parallel input and output interface with the CR710 ramp controller. It has the interface assignments shown in Table 3.

Appendix 15 provides a detailed description of interface functional assignment.

Each data modem logic level output is an uncommitted NPN transistor collector which when turned on sinks 2 mA at less than 1 V and which when turned off leaks less than 10 μA at 25 V. Each data modem logic level input is turned ON when the input voltage is less than 2 V; is turned OFF when the input current is less than 100 μA; does not place more than 20 V or any negative voltage on the inputs; and does not deliver more than 10 mA to a short circuit to d.c. common.

Type 170 Ramp Controller

The Type 170 ramp controllers were used in the 1980 system expansion. Type 170 is the standard MTC traffic signal controller. Reference 2 describes the Type 170 hardware function and specification. The Type 170 ramp controller software is based on Caltrans District 11 ramp metering software version 6.75 with modifications made by MTC on the communication interface and the ramp queue override function.

The software allows the 170 controller to control two ramps at the same time. Thus, there is only one 170 controller at each new interchange expansion (i.e., Cawthra Road, Winston Churchill, and Ford Drive interchanges). The remote ramp control command bit will be transmitted from the traffic control computer to the 170 controller through input port 5. The 170 controller will return the ramp control command status to the traffic control computer with a similar command bit pattern arrangement through output port 3. Since one 170 controller will control two ramps, the traffic control computer will normally alternate between ramp 1 and ramp 2 remote control command. Table 4 shows the 170 remote ramp control command definition.

The ramp queue override function is similar to the function implemented by the central traffic control computer.

2.23/ Changeable Message Sign Controller Unit and Interface

The changeable message sign (CMS) used in the QEW FSCS is a very low power consumption, high visibility, high reliability, rotation coloured magnetic disc type display. The CMS station which is installed upstream of the Southdown Road interchange, consists of a sign display, and a sign controller with an RS232C interface which is connected to a sign data modem. The sign display consists of two lines of 22 characters (modules) per line display. Each module consists of 5 x 7 rotating disc display elements. The typical viewing distance is 180 m (600 feet) at a degree from normal viewing angle. This provides approximately 7 s of viewing time, ample for motorists travelling at 100 km/h. Appendix 16 shows the sample sign display and format. The sign controller's serial communication input and output interface operates at a data rate of 300 baud. It accepts serial 10-bit ASCII character data which consists of 1 start bit, 8 data bits with even parity check bit, and 1 stop bit. Appendix 17 shows the sign display character symbol and its corresponding ASCII code. The CMS controller recognizes a fixed format message which consists of an address portion and a data portion from the traffic control computer. Appendix 18 shows the CMS controller input message format. In addition, the CMS controller has the capability to display the lines of a message in any order and to echo back each received character for verification purposes. In the event of power failure, an automatic blankout feature provides protection against the display of incorrect messages. The blankout feature will also be activated when the CMS controller receives a blankout control character.

When the CMS controller exercises the blankout feature, a "Break" signal will be presented on the echo signal line from the CMS controller to the CMS data modem. Appendix 19 shows the interface connector pin assignments between the CMS controller and the CMS data modem.

3/ QEW FSCS REAL-TIME SOFTWARE DESCRIPTION

The QEW FSCS real-time software is running under the Mapped Real-Time Disk Operating System (MRDOS) on the NOVA 4/X computer with 256KB of main memory and 10MB of disk storage space. The NOVA 4/X traffic control computer is the master decision-maker in the overall system. Through a front-end communication processor HP 21MX computer, the traffic control computer communicates with the field traffic devices and exercises the traffic surveillance and control functions. The following section describes the overall system operation.

3.1/ System Operation Overview

The overall system organization is a three-level hierarchical structure. The first level consists of the ramp controllers, loop detectors, and changeable message sign (CMS) subsystems which are installed in the field. The second level consists of the front-end communication processor and data modems which are linked to the ramp controllers, loop detectors, and CMS controller through the coaxial cable network. The function of the communication processor is to handle the real-time vehicle detector signal sampling task, and to command the ramp controllers and the CMS controller according to the controller command message transmitted from the traffic control computer. The third level consists of the traffic control computer system which is connected to the communication processor through a duplex data channel interface. The traffic control computer, which exercise the co-ordinated ramp control logic, incident detection logic, and CMS control logic is the master decision-maker in the whole QEW FSCS system. Dynamic freeway traffic information is collected by the loop detectors and transmitted to the front-end communication processor through a dedicated high-speed (48 000 bits per second) time division multiplexing data communication system. The communication processor preprocesses the freeway traffic data and presents the traffic data, in terms of vehicle signal elapsed time and volume to the traffic control computer at a rate of up to once per second. The traffic control computer then assesses the system-wide information about traffic conditions and responds to meet the changing traffic behaviour by sending ramp controller signal cycle length commands and a CMS controller sign message to the communication processor. The communication processor then transmits the signal cycle length commands and the CMS sign message to the corresponding ramp controller and CMS controller to adjust the freeway traffic pattern. The ramp controller also has the capability to operate independently in a time-of-day clock mode or mainline occupancy actuated mode. The communication processor has the stand-alone CMS control capability. Thus, any failure in the central traffic control computer system will not cause a complete stoppage of system operation. The various real-time FSCS software programs are shown as a simplified functional block diagram in Appendix 20. The programs are written in real-time FORTRAN 4 and assembler languages and are under control of a master task-scheduling program.

3.2/ FSCS Real-Time Task Analysis

In general, the real-time FSCS software has to perform the following basic tasks:

1/ Sampling and processing of traffic detector data pulses
2/ Smoothing and accumulation of the traffic detector data
3/ Freeway incident detection
4/ Ramp metering signal control and monitoring
5/ Changeable message sign control and monitoring
6/ Field hardware status monitoring
7/ System operator command interpreting
8/ Traffic system status displaying
9/ Data communication

The traffic detector data processing and hardware status monitoring tasks require the inputting of digital signal values and the updating of these values at fixed, precise intervals in order to provide the data for other traffic control and analysis task programs. These intervals range from fractions of a second to minutes, and hours. Although this periodic interval length is large compared to the response time (in microseconds) of a digital computer, the number of variables (hundred-to-thousands) involved is large enough so that these task programs must be very efficient in order to avoid using an excessive fraction of computer capability and to ensure that the data is updated precisely at the correct time.

Similar requirements are also present in the freeway incident detection, the ramp metering signal control, and the changeable message sign control tasks which use traffic process data and a number of filtering and control algorithms in order to calculate the appropriate output to be transmitted to the ramp controllers and CMS controller. These control algorithms and their associated outputs must be recalculated and updated periodically at approximately the same rate as the traffic detector data acquisition tasks and with the same precision.

Operation of the system operator command interpreting task and the traffic system status displaying task may be periodic (as in the display of traffic data gathered by the traffic data acquisition task) but it is also often required to provide fast response to random demand (e.g., when the system operator is attempting to communicate with the traffic control programs in order to change their operation or parameters). In short, the real-time FSCS software is concerned with a large number of surveillance and control tasks, each of which is repeated rather often with strict real-time response constraints.

In the QEW FSCS, Task 1 is carried out by the HP 21MX communication computer at 1/25 second intervals. Task 5 can also be carried out by the communication processor when operating in the stand-alone mode. Tasks 2 to 8 are carried out by the traffic control computer. Task 9 requires the co-operation of the communication computer and the traffic control computer.

3.3/ Mapped Real-Time Disk Operating System

The operating system used for the QEW FSCS software is the Mapped Real-Time Disk Operating System (MRDOS) which operates on the NOVA 4/X traffic control computer. The MRDOS is a vendor-supplied software package which will create the real-time multi-task programming environment for the FSCS application software. When initiated by real-time FSCS software, the MRDOS operating system will provide real-time task scheduling, memory management and protection, program overlay, input-output management, data file management, foreground and background programming and other functions required for real-time multi-task operation.

Since the MRDOS is oriented towards the computer hardware, it has to be configured and generated according to the QEW FSCS software requirement. This procedure is called the MRDOS system generation procedure and is well documented by the vendor. The other software support packages which operate under MRDOS are the FORTRAN 4 compiler, assembler, editor, MRDOS command language interpreter, relocatable loader, FORTRAN 4 run-time library, etc.

3.4/ FSCS Real-Time Application Software

The QEW FSCS real-time application software is oriented toward freeway traffic surveillance and control application. The application software is task-oriented, modular in structure, and designed mainly in real-time FORTRAN high-level language for its flexibility, and run under the MRDOS operating system. There are 69 programs in the QEW FSCS real-time application software system. These 69 programs are organized into five real-time tasks and require three overlay nodes in the logical memory address space. The MRDOS relocatable loader generates the FSCS real-time software save file and overlay file which contains all three overlay nodes disk resident program files. Besides the 69 program files, the FSCS real-time software system also contains seven-colour graphic display support data files, two CMS message files, and four ramp-metering operation support files. When in operation, the FSCS real-time software system will produce eight traffic and system operation record files for the purpose of off-line analysis . The FSCS real-time software system's main memory resident programs occupy approximately 44K bytes of memory and its disk resident programs occupy approximately 74K bytes. To run the QEW FSCS real-time software system requires 64K bytes of user logical memory address space, of which 44K bytes are for main memory resident programs and 20K bytes are for run time data buffers. In addition, approximately another 64K bytes of system logical memory address space are required to house the MRDOS software package. Appendix 21 shows the real-time FSCS application software program overlay structure and program size. The following program description is based mainly on the five real-time FSCS program tasks structure. The five real-time tasks organization is developed to satisfy both the periodic nature of the surveillance and control functions and the external asynchronous random demand input. These five real-time tasks are as follows:

- QEW -- System real-time task scheduling and operator command interpreter program
- COMCOM -- One-second communication and data processing program
- TRANSF -- Thirty-second data acquisition and control program
- GMAP -- One-minute colour graphic system status display program
- ALMSGN -- One-minute dummy task program reserved for future corridor control communication usage.

Each task program calls various numbers of subroutines to accomplish the designed FSCS functions.

3.5/ System Task Scheduling Program

The QEW FSCS real-time software system is activated by the system task scheduling program called QEW, which initializes, schedules, and controls all the real-time traffic surveillance and control task programs in a precise, ordered manner for real-time traffic management purposes. In particular, it performs the following tasks:

1/ It resets system hardware status.

2/ It opens overlay file "QEW.OL".

3/ It calls subroutine "PASSWORD" to query operator for the appropriate system start-up information.

4/ It calls subroutine "INITIALIZE" to initialize system parameters, to activate the task programs "COMCOM", "ALMSGN", "TRANSF" and "GMAP" in the order of descending priority level, to create or to recover FSCS data files.

5/ It calls subroutine "HEADING" to query operator for weather information, to check system real-time clock and align the start-up time at the end of the minute, to display system identification and start-up time message and to create weather record file "WEATHER".

6/ It calls subroutine "IBDEF" to identify communication processor to the MRDOS for computer-to-computer communication usage.

7/ It calls subroutine "CLOCK" to request a one-second real-time clock interrupt from the MRDOS as the input to the QEW FSCS software system.

8/ It then resets its own execution priority to the lowest level.

9/ The system control is then passed on to the operator command interpreter program "TOPCOM" and remain in "TOPCOM" until the operator issues the command to stop the system execution. At that time, the system control is returned to "QEW" and the program stops the execution. The system control is then returned to the MRDOS. Appendix 22 shows the real-time multi-task programs scheduling sequence.

3.6/ Communication Program

This task program "COMCOM" is to poll the front-end communication processor at one-second intervals for the most recent (real-time) traffic and field hardware status data and to transmit the appropriate ramp metering and CMS sign control commands to the communication processor through the duplex DMA data channel. Data communication is accomplished through the execution of different protocols that are designed to accommodate different data transmission needs. The communication processor acts as a messenger and relays the data between the traffic control computer and the field hardware devices.

Once activated by the "INITIALIZE" routine, the "COMCOM" task program becomes active and immediately suspends itself by executing the system task synchronization routine "CALL REC". From then on, this task program becomes active only when "CLOCK" routine executes the corresponding system task synchronization routine "CALL XMT". Specifically, the "COMCOM" program performs the following tasks:

1/ It suspends itself and waits for an activation signal from the one-second clock interrupt routine "CLOCK".

2/ When activated by the "CLOCK" routine, it executes the protocol 1 to receive 545-word traffic data buffer "DETECT" from the communication processor. Appendix 23 shows the "DETECT" traffic data buffer format.

3/ It calls subroutine "DATAIN" to process one-second volume, occupancy and speed data.

4/ It calls subroutine "MODM1" to accumulate modem response error counts and to disable or enable detectors based on data modem status.

5/ It calls subroutine "RAMCON" to prepare ramp metering control command buffer "COMDOP" and to check transmission error.

6/ It executes protocol 2 to transmit ramp metering control commands to the communication processor.

7/ Every 10 min, it executes protocol 3 to receive modem response count through "MODEMT" buffer.

8/ When requested by the operator through flag word "P4FLG", it executes protocols 4 and 5 to transmit and receive CMS messages through buffers "XMTBUF" and "RCVBUF".

9/ It goes back to step 1 waiting for the activation signal from "CLOCK" routine one second later. Appendix 24 shows the "COMDOP" ramp control command buffer format.

3.7/ Traffic Surveillance and Control Program

This task program accumulates and computes the real-time traffic data and saves the data files on the system data bank disk. It also activates the incident detection program, ramp control program, and hardware monitoring program. The traffic data base created and managed by this program contains both lane and station values of volume, occupancy, speed, and long vehicle count, each updated at 30-s, 1-min and 5-min intervals. The accumulation of lane volume, occupancy, vehicle speed, and long vehicle count is done continuously when the system is in operation. The entire traffic data base is disk resident and is divided into three groups as follows:

- Group 1 -- 30-s volume, occupancy, speeds, long vehicle count by lane
- Group 2 -- 30-s volume, occupancy, speed, by station
  -- 1-min volume, occupancy by station updated every 30 s
- Group 3 -- 5-min volume, occupancy, speed, long vehicle count by lane

The programs required to create and utilize the preceding traffic data base are described as follows.

(1) TRANSF Program

This is the main 30-s traffic surveillance and control task program. It performs the following tasks by calling various subroutines from the system overlay file QEW.OL every 30 s:

(1.1) It suspends itself and waits for an activation signal from the one-second task subroutine "DATAIN".

(1.2) It obtains data and time information from the system clock subroutine.

(1.3) It calls overlay subroutine "UPD30S" to process 30-s time slice traffic data.

(1.4) One minute after the system has started, it calls overlay subroutine "INCDET" to execute mainline station incident detection.

(1.5) After calling "INCDET" subroutine, it calls overlay subroutine "PERCH" every 30 s to perform station incident detection 1-min persistance check algorithm.

(1.6) It then calls overlay subroutine "DATST" every 30 s to extract 30-s time slice mainline lane data for freeway lane incident detection algorithm.

(1.7) It calls overlay subroutine "CLDU" every 30 s, to execute first-layer committee machine algorithm for freeway lane incident detection.

(1.8) It calls overlay subroutine "VTLU" every 30 s to execute second-layer and third-layer committee machine algorithm for freeway lane incident detection.

(1.9) It then calls overlay subroutine "RCTIME" every 30 s to execute co-ordinated traffic responsive freeway ramp metering control algorithm.

(1.10) It calls overlay subroutine "QUOCC" every 30 s to exercise balanced ramp queue override control algorithm.

(1.11) It calls overlay subroutine "ALM30" every 30 s to check field hardware device status, i.e., double loop detectors, ramp metering contollers, ramp metering signal lights, and changeable message sign controller.

(1.12) It calls overlay subroutine "ALM5M" every 5 min to check loop detector failure on 5-min time slice and 15-min time slice basis.

(1.13) It calls overlay subroutine "AMODM" every 10 min to check data modem hardware and to output date and time message on the system console printer.

(1.14) It calls overlay subroutines "DIS1" and "DYS1" every 30 s to merge 30-s time slice data into 1-min time slice data for colour graphic display file "QEWDYN".

(1.15) It calls overlay subroutine "DIS2" every minute to update 1-min time slice ramp control and mainline station data on colour graphic display file "QEWDYN".

(1.16) It activates colour graphic display task program "GMAP" every minute.

(1.17) It sends the beep sound every 30 s to the system keyboard terminal when there is a hardware device alarm signal.

(1.18) It then goes back to step 1.1, waiting for a task activation signal from "DATAIN" 30 s later.

3.7.1/ Traffic Data Acquisition Function

The traffic data acquisition function is accomplished jointly by a one-second program "DATAIN", and the 30-second programs "UPD30S", "DIS1", "DISY" and "DIS2"

(1) DATAIN Program

This program is called by one-second task program "COMCOM" after obtaining detector data buffer from the HP 21MX communication computer. Along with SLCHEK, DLCHEK and NXFER routines, the program performs the following tasks:

(1.1) It checks detector data in the "DETECT" buffer for logical consistency like:
- 0 < volume < MAXVEH (=4)
- Accumulated occupancy/volume > IMIN (=3)
- Accumulated double elapsed time/volume > IMIN (=3)

(1.2) It rejects out-of-range data and accumulates valid data every second.

(1.3) It processes speed sampling loop detector data, taking lane change into account.

(1.4) It stores ramp metering signal lights status for each metered on-ramp.

(1.5) At the end of every 30 s, it calls NXFER routine to transfer accumulated data to 30-s working buffers VSPEED, DATA30, and LIGHTS and reinitializes the one-second buffers.

(1.6) It then activates 30-s task program TRANSF.

(2) UPD30S Program

Basically this routine consolidates three-lane data into a station data. If data are not available in one lane, then station data are computed using the data from the other two lanes. The program performs the following tasks:

(2.1) It checks the validity of 30-s traffic data and sets to -1 if the data are found faulty. For single-loop detectors, only volume and occupancy are checked. For double-loop detectors, speed and vehicle-type data are OK only if both upstream and downstream detectors are OK.

(2.2) It consolidates lane data into 30-s and 1-min station data. Station volume and occupancy counts are in terms of three-lane equivalent counts. Station data set to -1 if all lane data found faulty.

(2.3) It computes section travel time, which is obtained as the ratio of section distance to the average speed of the upstream and the downstream stations.

(2.4) It writes the 30-s data to the disk file "DATA30" every 30 s.

(2.5) It accumulates 5-min data and writes to the disk file "DATA5M" every 30 s.

(3) "DIS1" Program

This program updates part of the colour graphic display file "QEWDYN" every 30 s. It performs the following tasks:

(3.1) It opens "QEWDYN" file and initializes local buffer QEWDYN.

(3.2) It defines number of queue detectors and exit detectors for each ramp in the array IDNO(10) and stores data and time information in the QEWDYN buffer. The most significant digit of each element of the array IDNO(10) contains the number of queue detectors for that ramp, and the least significant digit contains the number of exit detectors.

(3.3) It increments a 30-s counter to determine if it is the end of the minute. If not, it processes the current 30-s single loop detector data and mainline data in the upper part buffer of "QEWDYN" and store them in disk file "QEWDYN".

(3.4) It closes disk file "QEWDYN".

(4) DISY Program

At the end of the minute, this program performs the following tasks:

(4.1) It reads the preceding 30-s data from the disk file "QEWDYN" and integrates with the present 30-s data into 1-min data. The 1-min data includes 1-min ramp detector volume, occupancy, 1-min mainline lane volume, occupancy, and speed data.

(4.2) It writes the 1-min data to the disk file "QEWDYN".

(5) DIS2 Program

At the end of the minute, this program performs the following tasks:

(5.1) It reads ramp controller cycle length from "CYCLES".

(5.2) It updates mainline control mode, cycle length, ramp waiting time and smoothed queue detector occupancy on the buffer "QEWDYN".

(5.3) It updates station capacity, speed, volume, and occupancy (1-min data) and incident status.

(5.4) It writes to the disk file "QEWDYN" and records file number to disk file "DMPREC".

3.7.2/ Station Incident Detection Function

This program is called every 30 s to detect freeway station incidents. It consists of two routines, INCDET and PERCHK, and performs the following tasks:

(1) It computes four incident test values which are rounded off in percentage assuming 25 samples per second. The four test values are as follows:

- $OCC_{i,t}$ = occupancy at detector station i at time t
- $(OCC_{i,t} - OCC_{i+1,t})$ the occupancy difference at time, t, between the station, i, being tested and its downstream station, i+1.
- $(OCC_{i,t} - OCC_{i+1,t}) / OCC_{i,t}$ = the relative percentage occupancy difference between the station, i, and its downstream station, i+1, at time, t.
- $(AVGOCC - OCC_{i+1,t}) / AVGOCC$ = the relative percentage occupancy difference between time, t, and the average 30-s occupancy AVGOCC, at the downstream station, i+1.

Where $AVGOCC = (OCC_{i+1,t-30} + OCC_{i+1,t}) / 2$.

(2) It checks incident occurrence at the station, i, only if station i is cleared for any incident at time, t. An incident is detected, if all the four test values exceed the four threshold values at the station, i.

(3) It checks for the incident clearance at the station, i, only if there is a true incident at station, i, at time, t. An incident is cleared, if any one of the following conditions is satisfied:

$(OCC_{i,t} - OCC_{i+1,t}) / OCC_{i,t}$ less than threshold value, and $OCC_{i,t}$

less than $OCC_{i,Ti}$ and $OCC_{i+1,t}$ larger than $OCC_{i+1,Ti}$(where T_i

indicates the time preceding the occurrence of incident).

(4) It performs 1-min incident persistency check for the incident detected. The incident detected is regarded as "true" if it lasts at least 1 min.

(5) The incident data are written to disk file "INCLOG" for any true incident when it occurs as well as when it is cleared. The maximum mainline queue length is also computed and stored.

3.7.3/ Lane Incident Detection Function

This program is called every 30 s to detect freeway lane incident after station incident has been identified. The logic is based on Reference 3. It consists of three routines, DATSTR, CLDU, VTLU and performs the following tasks:

DATSTR

- It determines if a true incident has occurred in any station, i.
- It then obtains preceding 1-min logical records from disk file "DATA30" which stores 30-s time slice data.
- It retrieves lane occupancy, lane volume and lane speed data and stores in the buffer "DATAX".
- It obtains average station occupancy, station volume and station speed and stores in "DATAX".
- It writes "DATAX" buffer to disk as logical record in "DATAX" file.

CLDU

This routine implements a committee logic decision unit process. The routine selects lanes based on decision parameters occupancy, volume and speed which are selected from three 30-s time slice data.

(1) It checks through stations 1 to 21 for new station incident indication and skips station 3 which is located on Highway 403.

(2) If a station incident is found at station, i, it reads DATAX file for the station i and i+1 data and lane data.

(3) It then implements the committee logic decision unit based on the occupancy parameter according to the following process:

(3.1) It selects the lane, n, with the minimum lane occupancy at the downstream station, i+1, at time slice T.

(3.2) It computes:

$$O_n(T) = (AVGOCC_{T,i+1} - MINOCC_{T,i+1}) / AVGOCC_{T,i+1}$$

where: $MINOCC_{T,i+1}$ = minimum lane occupancy at lane, n, of downstream station, i+1, at time T.

$AVGOCC_{T,i+1}$ = average downstream station occupancy at time, T.

(3.3) If $O_n(T)$ is larger than the threshold value, the incident is detected at lane, n.

(3.4) If $O_n(T)$ is less than the threshold value, then the lane at the station i with the maximum lane occupancy is selected as the incident lane.

(3.5) It stores the incident lane information in the file "RESDEV".

(3.6) It repeats the steps (3.1) to (3.5) for time slices T-30, and T-60.

(4) It implements the committee logic decision unit based on the volume parameter according to the following process.

(4.1) It selects the lane, n, with the minimum lane volume at the downstream station, i+1, at time slice, T.

(4.2) It computes:

$$V_n(T) = (AVGVOL_{T,i+1} - MINVOL_{T,i+1}) / AVGVOL_{T,i+1}$$

where: $\text{MINVOL}_{T,i+1}$ = minimum lane volume at lane, n, of downstream station, i+1 at time T.

$\text{AVGVOL}_{T,i+1}$ = average downstream station volume at time, T.

(4.3) If $V_n(T)$ larger than the threshold value, then the incident is detected at lane n.

(4.4) If $V_n(T)$ less than the threshold value, then the lane at the station i with the minimum lane volume is selected as the incident lane.

(4.5) It stores the incident lane information in the file "RESDEV".

(4.6) It repeats steps (4.1) to (4.5) for time slices T-30 and T-60.

(5) It implements the committee logic decision unit based on the speed parameter according to the following process:

(5.1) It selects the lane, n, with the minimum lane speed at the station, i, at time slice T.

(5.2) It computes

$\int_n(T) = (\text{AVGSPD}_{T,i} - \text{MINSPD}_{T,i}) / \text{AVGSPD}_{T,i}$

where

$\text{MINSPD}_{T,i}$ = minimum lane speed at lane, n, of the station, i, at time T.

$\text{AVGSPED}_{T,i}$ = average station speed, at time T

(5.3) IF $\int_n(T)$ larger than the threshold value, then the incident is detected at lane, n.

(5.4) If $\int_n(T)$ less than the threshold value, then the lane at the downstream station, i+1, with the maximum lane speed is selected as the incident lane.

(5.5) It stores the incident information in the file "RESDEV".

(5.6) It repeats the process for time slice T-30, and T-60.

(6) It deletes the file "DATAX".

VTLU

This routine implements a vote-taking logic unit. The routine counts the lane selection (i.e., votes) from the CLDUs and makes the decision based on the majority rule.

(1) It checks through stations 1 to 21 for a new incident detected and skips station 3.

(2) If an incident is found at station, i, it reads "RESDEV" file to obtain three incident lane selections made by each of the CLDUs.

(3) It compares lane selections made by occupancy CLDU.

(4) If any two agree, they automatically form a two-out-of-three majority. The outcome (i.e., decision) is set to the value of the comparison pair which gives the majority ruling.

(5) If no two agree, then all are different and a three-way tie has resulted. The outcome is reset to 0, and considered a null vote which will not be counted.

(6) It repeats steps 3 to 5 with lane selection result from volume CLDU.

(7) It repeats steps 3 to 5 with lane selection result from speed CLDU.

(8) It modifies the lane vote total from occupancy CLDU, by applying occupancy weighting factor.

(9) It modifies the lane vote total from volume CLDU, by applying volume weighting factor.

(10) It modifies the lane vote total from speed CLDU by applying speed weighting factor.

(11) It then chooses the lane with the maximum number of votes as the lane of incident.

(12) It writes lanes incident data to "INCLOG" and "RESDEV" files.

3.7.4/ Ramp Control Function

The ramp control function is accomplished jointly by one-second program RAMCON, and 30-s programs RCTIME, and QUOCC. The RCTIME and QUOCC programs implement a system control strategy which is an on-line real-time process. The process calculates a new metering strategy every 30 s according to existing traffic conditions. Since the calculation is made each 30 s to determine the metering strategy for the next 30 s, the feedback from the metering strategy is for all practical purposes a continuous operation. Under this system control strategy, a given entrance ramp to be metered would have upstream, local, and downstream detector stations assigned to provide upstream station volume, local station occupancy, and downstream station occupancy data. The strategy would compare each of these three parameter measurements against a table of threshold volume and occupancy values, and pre-assigned metering rates would be obtained for each of these measurements.

The smallest of the determined metering rates is then selected. In order to balance the ramp queue length and mainline traffic condition, a ramp queue override feature is implemented by the QUOCC program. When the ramp queue override feature is not activated, the entrance ramp would meter at the selected minimum metering rate over the next 30 s, at which time the parameter's measurement would again be evaluated for the following 30-s metering rate, and so on. If the ramp queue override feature is activated, the selected minimum metering rate may be replaced by a higher metering rate picked by the QUOCC program for the next 30 s. This control process will be repeated every 30 s. Appendix 25 describes the computerized ramp metering rate selection logic and Appendix 26 describes the ramp queue detection logic and ramp queue override feature.

RCTIME

This program selects the ramp metering rate (cycle length) based on the mainline traffic condition.

(1) It obtains assigned mainline detector station number for each entrance ramp from disk file "RAMTAB".

(2) It obtains threshold values of occupancy for the local and downstream station, the threshold value of volume for the upstream station, and the corresponding cycle length for each of six different metering threshold values from disk file "CYCTAB".

(3) It computes control period, and by default, automatic advancement in control period is implemented.

(4) If the control period equal to "0", then:

  (4.1) It assigns permanent green to those ramps not under manual command override.

  (4.2) For those ramps under manual command override, it leaves them as they are.

  (4.3) It returns the program control to the TRANSF program.

(5) If the control period is equal to "1", "2", or "3", then:

  (5.1) It ignores those ramps dropped offline, i.e., in non-metering condition, during control period 3, or under ramp queue override condition, or under manual command override condition.

  (5.2) For each ramp to be controlled according to the mainline traffic condition, it locates the corresponding mainline detector station number and volume column number from table "RAMTAB".

(5.3) It then derives the cycle length based on the comparison between the mainline detector station occupancy, volume and the threshold values stored in table "CYCTAB". The mainline detector station data used for metering rate (or cycle length) selection will be the latest 1-min running average occupancy by station and the latest 1-min running average volume by station. These 1-min running average volume and occupancy data are updated every 30 s. The cycle lengths derived range from no metering (green) to cycle length number 1 to 5. In control period 1, once a ramp starts metering, it will be metered at least with minimum cycle length number (maximum metering rate). In control period 2, all ramps are metered at least with minimum cycle length number. Three different cycle length numbers are selected by comparing mainline downstream and local occupancy and upstream volume with their corresponding threshold values. The cycle length chosen is the largest of the three and is stored in the RAMCYC buffer.

(5.4) It then determines the ramp metering control mode and stores in the MODCYC buffer.

QUOCC

This program modifies, if necessary, the ramp metering rate (cycle length) selected by the RCTIME program based on the ramp queue length condition.

(1) It obtains ramp queue length vs ramp queue detector occupancy table and ramp controller cycle length table from disk file QUETAB and CYCLES.

(2) It smooths 30-s ramp queue detector occupancy values.

(3) It initializes ramp queue lengths and ramp waiting time to 0.

(4) It bypasses step 5 for each ramp if:
- there is no metering condition in control period 1 or 3,

- the ramp controller is in local control mode (RAMCYC=6), or
- the control period is 0.

(5) For each ramp,

- It computes ramp queue length. Mississauga S/B has ramp queue length equal to the sum of the ramp queue lengths derived from the two queue detectors; Erin Mills and Ford Drive have queue length equal to the maximum of the two queue lengths. The ramp queue length and ramp waiting time is set to -1 if queue detector fails.

- It modifies ramp metering cycle length based on ramp queue override logic. If an entrance ramp is not under manual command override or ramp queue override logic function is not disabled by operator, ramp queue override logic is "on" when the ramp queue detector occupancy value exceeds the upper bound threshold value and "off" when it falls below the lower bound threshold logic.

- It computes ramp waiting time.

RAMCON

This program is activated by task program COMCOM as a one-second routine to transmit the ramp controller cycle length command to the front-end communication computer which will relay the command to the field ramp controller.

(1) It looks for manual cycle length override commands entered by the operator or cycle length command generated by RCTIME or QUOCC.

(2) It translates cycle length command into command bit pattern and stores in command buffer COMDOP. For dual ramps which are controlled by one type 170 controller, control commands are transmitted every 2 s per ramp. For a single ramp which is controlled by one CR710 controller, commands are transmitted every second.

(3) It stores both transmitted control commands and received controller status for each ramp.

(4) It checks transmission errors for both types of controllers, CR710 and type 170. If the number of errors exceeds 30 consecutive times, the status information bits will be saved in the CHEK1 buffer and the command information bits will be saved in the CHEK2 buffer.

3.7.5/ Field Hardware Monitoring Function

The field hardware monitoring function is accomplished by one-second routine MODM1, 30-s routine ALM30S, 5-min routine ALM5M, and 10-min routine AMODM.

MODM1
This routine is activated every second by the task program COMCOM.

(1) It checks and accumulates data modem response counts from buffer DETECT. If the response counts differ from the nominal count (25) by three (3) or more, error counts are accumulated in the buffer MODEM1.

(2) It disables detectors connected to the data modem which does not respond five consecutive times.

(3) It enables detectors connected to the data modem which responds five consecutive times after it failed.

(4) It increments a 10-min time counter for activation of AMODM program.

(5) It transfers data modem error counts to the buffer MODEMH (MODEMH(1) = MODEMT(152)) every 10 min.

ALM30S
This routine is activated every 30 s by the task program TRANSF to monitor mainline detector, ramp controller, signal light and CMS hardware status.

(1) It checks mainline detectors for malfunction when the detector station speed is greater than 14 m/s (approximately 50 km/h). Under

this condition, a mainline detector is considered faulty if the 30-s detector occupancy value exceeds 95%.

(2) It checks the command data transmitted by the traffic control computer against the echo-back status from the ramp controller.

If the command data bit is inconsistent, the error bit is recorded in the buffer CSTAT array.

(3) It checks the ramp metering signal light according to the following conditions:

(3.1) When RAMCYC(I) = 0, the corresponding entrance ramp should have the green light ON.

(3.2) All green, yellow and red lights should be ON at least once during the past 30 s for a normal metering cycle (RAMCYC(I) ≠ 0); if not, the demand detector volume must be 0. An error message is delivered to the operator only if three consecutive errors were found. The light status error is recorded in the buffer LTSTAT.

(4) It records sign malfunction information to disk file ALMREC when notified by TP14D through flag word ALMFLG.

(5) It records any hardware malfunction to disk file ALMREC and sounds a beep at dasher terminal to notify the operator.

ALM5M

This program is activated every 5 min by the task program TRANSF to monitor detector hardware operation.

(1) It obtains 5-min time slice detector data from disk file DATA5M.

(2) It accumulates and checks 15-min exit ramp detector volume. If the 15-min volume is less than threshold value (EXVOL = 1), the exit detector is considered failed.

(3) It checks single loop and mainline double loop detectors according to the following conditions:

(3.1) for demand detector -- 5-min occupancy > 1%
(3.2) for exit detector -- 5-min occupancy < 95% or
15-min volume > 1
(3.3) for passage, queue, and mainline detector:
-- 1% < 5-min occupancy < 95% and
-- 1 < 5-min volume < 250

(4) Any detector malfunction is recorded in the disk file ALMREC and an alarm beep is sounded at the dasher terminal.

AMODM

This program is activated every 10 min by the task program TRANSF to monitor data modem malfunction.

(1) It prints date and time information on the dasher terminal.

(2) It checks data modem for malfunction from buffers MODEMT and MODEM, and records any malfunction for a maximum of 10 consecutive times for the same data modem to the disk file ALMREC.

(2.1) Specifically, the one-second modem response count error in the buffer MODEMH is checked.

(2.2) The modem address responses in the buffer MDMTMP, which are obtained from communication protocol 3, are checked for valid modem address in each corresponding buffer location.

(3) If CMS sign data modem fails, "SIGN MODEM MALFUNCTION" is printed out at dasher terminal.

3.8/ Traffic Display Control Task Program

This task program controls the colour graphic display terminal (Tektronix 4027) and provides a dynamic traffic situation display on the colour graphic CRT terminal screen. The traffic situation display can be coloured based on either traffic speed or occupancy information. Based on this real-time traffic situation display, the operator can take the necessary actions -- for example, he can zoom in the TV camera on a particular spot, phone the police, override the ramp metering cycle length or send the traffic advisory message to the changeable message sign. The programs required to control the colour graphic display terminal are described below.

(1) **GMAP Program**

This is the main 1 min colour graphic traffic display control task program. It calls various subroutines to perform the following tasks:

(1.1) It suspends itself and waits for activation signal from the 30 s task program "TRANSF".

(1.2) It freezes the colour graphic traffic situation display if it is commanded by the operator.

(1.3) It calls overlay subroutine "REQEW" every minute to read the colour graphic traffic situation display file "QEWDYN".

(1.4) It calls overlay subroutine "CHQOR" every minute to check for ramp queue override condition on each entrance ramp.

(1.5) It calls overlay subroutine "ACCIDENT" every minute to check for freeway station incident status.

(1.6) It then opens the corresponding graphic polygon file for the map display requested by the operator. There are seven graphic polygon files corresponding to seven map displays; namely, the QEW corridor map, the Ford Drive interchange map, Winston Churchill interchange map, Erin Mills interchange map, Mississauga interchange map, Highway 10 interchange map, and Cawthra interchange map.

(1.7) If the operator requests the QEW corridor display, it calls overlay subroutine "QSPD" to display heading, date, time and then follows steps 1.8 to 1.18. If the operator requests the interchange display, it goes to step 1.19 directly.

(1.8) It calls overlay subroutine "QOCC" if the display colour code option is based on occupancy instead of speed.

(1.9) It reads the QEW corridor graphic polygon file "QPOLY" to display bridges.

(1.10) It calls overlay subroutine "RAMPS" to display vectored ramps configuration.

(1.11) It calls overlay subroutine "DETECTOR" to display mainline detector stations.

(1.12) It then reads the rest of "QPOLY" to display the QEW corridor street network.

(1.13) If an incident has been detected, it calls overlay subroutine "FACC" to obtain the first three incident locations and then follows steps 1.14 to 1.18. If there is no incident, it goes to step 1.16 directly.

(1.14) It calls overlay subroutine "FLASH" to write the graphic polygons for the incident location into a temporary file "TEMPFILE".

(1.15) It calls overlay subroutine "LOOP" to flash the incident location polygon on the colour graphic terminal screen by looping on "TEMPFILE".

(1.16) It then displays any changeable message sign hardware related message if there is any on the QEW corridor display.

(1.17) It then closes the "QPLOY" file and 4027 I/O channel.

(1.18) It then goes back to step 1.1 waiting for activation signal from "TRANSF" task program 1 min time slice later.

(1.19) It calls overlay subroutine "DISMAP" to initialize lane element, to determine number of mainline stations for display, to place heading, date and time.

(1.20) It calls overlay subroutine "SPD1" to determine what colour to paint the lane based upon the 1 min mainline lane speed data.

(1.21) It calls overlay subroutine "SPD2" to paint the driving lane and extra lane on the Winston Churchill interchange based on the traffic speed.

(1.22) It calls overlay subroutine "RAMPS" to colour the vectored ramps for the QEW corridor display and the ramp polygon for the interchange displays.

(1.23) It calls overlay subroutine "IOCC" to determine which colour to paint the interchange display lane if the display colour code option is based on 1 min mainline lane occupancy data.

(1.24) It calls overlay subroutine "IOC2" to paint the extra lane on the Winston Churchill interchange based on lane occupancy data.

(1.25) It calls overlay subroutine "RAMPS" to display interchange ramps as polygons.

(1.26) It calls overlay subroutine "BOX1" to read number of stations and controllers, to initialize speed, volume and occupancy elements, and to draw green boxes which contain the station speed, volume and occupancy value.

(1.27) It calls overlay subroutine "BOX2" to display mainline and ramp station speed, volume, and occupancy data.

(1.28) It calls overlay subroutine "BOX3" to display ramp metering cycle time for on-ramp and station 15A mainline data.

(1.29) It calls overlay subroutine "RDISMAP" to display miscellaneous polygons, detector stations, ramp metering arrows, colour code legend, textual data, and "North" direction pointer.

(1.30) It closes the 4027 colour graphics terminal I/O channel and colour graphic polygon files.

(1.31) It then goes back to step 1.1 waiting for task activation signal from "TRANSF" 1 min time slice later.

3.9/ Corridor Control and Surveillance Task Program

This task program is activated, at one-minute intervals by the main task scheduling program. This task is reserved for future corridor control communication usage.

3.10/ System Operator Control Command Function

The system operator control command function is accomplished by a set of overlay subroutines called by the TOPCOM program which is, in turn, called as the lowest priority program by the master task scheduling program QEW. There are seven command groups in the system to provide control commands for the operator. Table 5 shows the command functions and the overlay subroutine designed to accomplish the functions.

An input editor routine INPEDIT, is called by TOPCOM to accept and decode input commands. Depending on the command received, the control is passed to various subroutines for action.

3.11/ System I/O Channel Usage Assignment

Table 6 describes the I/O channel usage assignments chosen to avoid conflict in NOVA 4/X control computer I/O channel usage and to facilitate future change in the control software.

3.12/ System Parameter Definition

Table 7 defines the control parameters used in the QEW FSCS real-time control software system.

3.13/ QEW FSCS Program File Organization

The QEW FSCS real-time software system has 69 programs. The programs are stored in the disk files in three different formats. The source program file is identified by the suffix .SR and is used in the system development stage. The binary program file is identified by the suffix .RB and is generated by the compiler or assembler. The relocatable loader processes the binary files and generates the QEW save file identified by the suffix .SV and the QEW overlay file identified by the suffix .OL for execution purposes. When a system change is to be made, a source program is created or modified, then compiled, and loaded by the loader to create a new QEW save file and overlay file for implementation. Appendix 36 shows the real-time software system source program listing.

3.14/ QEW FSCS Data File Organization

The QEW FSCS real-time software system generates eight data files for system analysis purposes. The Appendices 27, 28, 29, 30, 31, 32, 33, and 34 show, respectively, the DATA30, DATA5M, QEWDYN, INCLOG, ALMREC, XMTTAB, RESDEV, and WEATHER files' logical record format. Table 8 shows each file's structure.

The first physical record of each file is used to identify file name, date of file creation, physical record length, number of physical records per logical record, logical record length, and the number of physical records. Appendix 35 shows the first physical record format of each file.

In addition, there are four ramp metering operation support files. Table 9 shows each file's structure.

There are also two CMS message files and seven colour graphic display support data files.

4/ QEW FSCS CONTROL COMMAND FUNCTION DESCRIPTION

Traffic Control Command is the man/machine interface between the traffic controller and traffic control computer system. The traffic controller relies on this man/machine interface facility to effectively manage and control a computerized freeway surveillance and control system (FSCS). To enhance this capability, a traffic control command definition meeting was held from 1978 10 25 to 1978 10 26 to review existing command performance and define new requirements. The following description is based on the past QEW FSCS experience and a distillation of the collective thinking expressed in the meeting.

Hierarchical Structure of Traffic Control Command

The Traffic Control Command will be represented by a three-level hierarchical tree structure. Figure 1 shows the first level of command structure.

QEW is the first-level on-line command. The command operates under MRDOS environment. This environment is indicated by a prompt character "R".

QEW Command

Under the "QEW" command, there are eight command groups which constitute the second level command structure. Figure 2 shows the second-level command structure.

These second-level commands operate under the QEW environment, i.e., under a real-time computer control environment. This environment is indicated by a prompt character ">". To assist the traffic controller in identifying each sub-command mode, a single "RETURN" key signal from the traffic controller will prompt the computer to print the sub-command name such as "RC", "QO", etc., on console TTY. To exit from any sub-command mode, an "E" character from console TTY will return the computer control to QEW" mode. In addition, all the commands are ended with a "RETURN" key signal from console TTY. A command escape character "$" (ESC key) is also available for use to abort or terminate any action command sequence and return the control to first-level command mode QEW.

4.1/ RC Second-level Command Group

Under "RC" second-level command, there are five action commands which are part of the third-level command structure. Figure 3 shows the third-level command structure under "RC".

(1) CN_1N_2..N_iM# Command

Format:

C N_1 N_2.. N_i M #

The command line consists of command name "C" and "M" followed by arguments N_1 N_2.. N_i and #, all separated by spaces.

Function:

The command is to override ramp controllers N_1 N_2.. N_i with cycle length #. Where:

N_1 N_2.. Ni = Controller identification number
# = Cycle length number

If only command name "C" is called for, then the cycle length of each ramp controller is displayed in an ascending order according to the controller identification number. There will be 10 virtual controllers in the expanded QEW FSCS.

Example:

```
> C 2 3 10 M 4
```

The command overrides ramp controller 2, 3, and 10 with cycle length 4

```
> C
2 3 4 1 5 3 4 1 5 3
```

The cycle length of each controller is displayed.

(2) PN Command

Format:

P N

The command line consists of command name P and argument N, separated by space.

Function:

"P N" command is to change advancement of control period from automatic mode to manual mode or vice-versa. Where:

N = 0, 1, 2, or 3 indicate manually selected control period
= 4 indicate a request for automatic advancement of control period

If only command name "P" is called for, then the present control period is displayed. If "CTRMOD" = 4, the output indicates "AUTO-ADVANCEMENT" instead of "MANUAL".

Example:

```
> P 3
```

To set ramp control period to 3.

```
> P
PRESENT CP = 3, MANUAL
```

To display present control period.

(3) $TN_1N_2N_3N_4$ Command

Format:

T N_1 N_2 N_3 N_4

The command line consists of command name T and arguments N_1, N_2, N_3 and N_4, all separated by spaces.

Function:

"T" command is to change ramp control period start time frame.
Where:

N_1 = start time for control period 1
N_2 = start time for control period 2
N_3 = start time for control period 3
N_4 = start time for control period 0

If only command name "T" is called for, then all the control periods time frames are displayed.

Example:

```
> T 655 715 815 900
```

To set the control period time frames to 6:55, 7:15, 8:15, and 9:00 for control period 1, 2, 3, and 0, respectively.

```
> T
CP START TIME:  655 715 815 900
```

To display control period time frames.

(4) R M N_0 N_1.. N_i Command

Format:

R M N_0 N_i.. N_i

The command line consists of command name R and arguments M, N_0, N_1, .., N_i, all separated by spaces. "i" value will depend on "M" value.

Function:

"R" command is to update the co-ordinated ramp control tables "RAMTAB", "CYCTAB", "CYCLES".

When: M = 1, the row elements of "RAMTAB" table will be updated

N_0 = the row number to be updated

N_1 N_2 N_3 N_4 = the new row element numbers

If only "R1" command is called for, then the 10 x 4 matrix element of "RAMTAB" will be displayed.

When: M = 2, the row elements of "CYCTAB" table will be updated

N_0 = the row number to be updated

N_1 N_2 N_3 N_4 N_5 = the new role element numbers

If only "R2" command is called for, then the 5 x 5 matrix element of "CYCTAB" will be displayed.

When: M = 3, the row elements of "CYCLES" will be updated

N_0 = the row numbers to be updated

$N_1\ N_2\ N_3\ N_4\ N_5$ = the new row element numbers

If only "R3" command is called for, then the 10 x 5 matrix element of "CYCLES" will be displayed.

Example:

```
> R 1 2 9 9 8 5
```

To update table "RAMTAB" row No. 2 with element 9, 9, 8, and 5

```
> R 2 3 50 60 66 78 87
```

To update table "CYCTAB" row No. 3 with element 50, 60, 66, 78, 87

```
> R 3 1 50 65 80 100 130
```

To update table "CYCLES" row No. 1 with element 50, 65, 80, 100, 130

(5) E Command

Format:

```
E
```

"E" command consists of command name E only.

Function:

"E" command is to exit from any second-level command mode and return to "QEW" command mode. The computer will print "QEW" on the console TTY to identify the "QEW" mode.

Example:

```
> E
QEW
>
```

4.2/ QO Second-Level Command Group

Under "QO" second-level command, there are seven action-commands which are part of the third-level command structure. Figure 4 shows the third-level command structure under "QO".

(1) N# Command

Format:

N #

"N" command consists of command name "N" followed by argument, #, separated by space.

Function:

"N" command is to enable ramp queue override logic on ramp No. #.

Example:

```
> N 3
```

To enable queue override logic on ramp No. 3.

(2) D# Command

Format:

D #

"D" command consists of command name "D" and argument, #, separated by space.

Function:

"D" command is to disable ramp queue override logic on ramp No. #.

Example:

```
> D 10
```

To disable queue override logic on ramp No. 10.

(3) L $N_1N_2 .. N_{10}$ Command

Format:

L N_1 N_2 .. N_{10}

The command line consists of command name L and arguments N_1, N_2, .., N_{10}, all separated by spaces.

Function:

"L" command is to update lower-bound Q-OCC threshold values on all 10 ramps in terms of number of pulses.

Where:

N_1 N_2 .. N_{10} = lower-bound values for 10 on-ramps

Example:

```
> L 600 600 600 600 600 600 600 600 600 600.
```

(4) U $N_1N_2..N_{10}$ Command

Format:

U N_1 N_2 .. N_{10}

The command line consists of command name U and arguments, N_1, N_2, .., N_{10} separated by spaces.

Function:

"U" command is to update upper-bound Q-OCC threshold values on all 10 ramps in terms of number of pulses.

Where:

N_1 N_2 .. N_{10} = upper-bound values for 10 on-ramps

Example:

```
> U 1050 1050 1050 1050 1200 1050 1050 1050 900 900
```

(5) TMSN_1N_2 .. N_{10} Command

Format:

T M S N_1 N_2 .. N_{10}

The command line consists of command names T and S, and arguments M and N_1, N_2, .., N_{10}, all separated by spaces.

Function:

The command is to set queue override time limit, M, before steady green and number of 30 s time step, N, during cycle length reduction process.

Where:

M = time limit in minutes

N_1 N_2 .. N_{10} = number of 30 s time step for all ten on-ramps

Example:

```
> T 20 S 6 6 6 6 6 6 6 6 6 6
```

To set queue override time limit to 20 min before steady green and time step to 3 min for all ten on-ramps.

(6) P Command

Format:

P

The command line consists of command name "P" only.

Function:

"P" command is to print out common buffer related to metering strategy and queue override strategy.

Example:

```
> P
```

```
-----------------------------------------------------------------------------
            RAMPS
-----------------------------------------------------------------------------
            No.1  No.2  No.3  No.4  No.5  No.6  No.7  No.8  No.9  No.10
-----------------------------------------------------------------------------
MODCYC         5     5     5     5     5     5     5     5     5     5
QWAIT          0     0     0     0     0     0     0     0     0     0
QOVER          1     0     0     0     0     0     0     0     0     0
QOCCLB       600   600   600   600   600   600   600   600   600   600
QOCCUB      1050  1050  1050  1050  1200  1050  1050  1050   900   900
QOCC          34   123    50    12   100   152    79    85    93    49
-----------------------------------------------------------------------------
```

(7) E Command

Same as Section 4.1 (5)

4.3/ CMS Second-Level Command

Under "CMS" second-level command, there are seven action-commands which are part of the third-level command structure. Figure 5 shows the third-level command structure under "CMS".

(1) A Command

Format:

```
A
```

The command line consists of command name "A" only.

Function:

"A" command is to add a message to the message file "MSGFILE".

Example:

```
> A
ROW 1, CR, ROW 2, CR
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
     I N C I D E N T  A H E A D
  D R I V I N G  L A N E  B L O C K E D
OK?
> Y
MSG No. 23
     I N C I D E N T  A H E A D
  D R I V I N G  L A N E  B L O C K E D
>
```

The computer also automatically assigns the message number 23 to the added message.

(2) B Command

Format:

```
B
```

The command line consists of command "B" only.

Function:

"B" command is to print out the content of message transmitting buffer and message receiving buffer.

Example:

```
   B
TRANSMITTING MESSAGE
MAINTENANCE OPERATIONS
   CENTRE LANE CLOSED
RECEIVING MESSAGE
```

```
MAINTENANCE OPERATIONS
   CENTRE LANE CLOSED
```

(3) D# Command

Format:

```
D #
```

The command line consists of command name "D" and argument #, separated by space.

Function:

"D" command is to delete a message with message number "#" from the message file.

Example:

```
> D 23
```

To delete message No. 23

(4) L Command

Format:

```
L
```

The command line consists of command name "L" only.

Function:

"L" command is to list all transmitted changeable message signs (CMS) message from message transmission file "XMTTAB". The messages listed will be classified according to their destinations, that is, CMS controllers.

Example:

```
> L
CMS 1.
7:50:30
MAINTENANCE OPERATIONS
   CENTRE LANE CLOSED
8:15:00
ACCIDENT AHEAD
```

RIGHT LANE CLOSED
CMS 2.
7:45:00
CONGESTION AHEAD
REDUCE SPEED

(5) S N_1 N_2 .. N_i M # Command

Format:

S N_1 N_2 .. N_i M #

The command line consists of command name "S" and "M" followed by arguments N_1 N_2 .. N_i and #, all separated by spaces.

Function:

"S" command is to send message with prestored message number "#", to the sign controllers which have identification number N_1 N_2 ..
N_i = sign controller identification number.

If only command name "S" and arguments "N_1 N_2 .. N_i" are called for, then the latest message transmitted to each sign controller identified shall be displayed in an ascending order according to the sign controller identification number.

Example:

S 1 2 M 3
FREEWAY OPEN
DRIVE CAREFULLY

This command shall send the message M 3 to sign controller 1 and 2, and shall also display the M 3 sign message.

S 1
FREEWAY OPEN
DRIVE CAREFULLY

This command shall display the latest sign message transmitted to the sign controller 1.

(6) P # Command

Format:

```
P #
```

The command line consists of command name "P" and argument "#", separated by space.

Function:

"P" command is to print a message with message number "#"

Example:

```
> P 5
MESSAGE No. 5
MAINTENANCE OPERATIONS
   CENTRE LANE CLOSED
```

(7) **E Command**

Same as Section 4.1 (5)

4.4/ G Second-Level Command Group

Under "G" second-level command, there are nine third-level action commands. Figure 6 shows the third-level command structure under "G". The "G" command group is to change graphic displays.

(1) **Q, F, W, S, M, H, C, N, E, O, V Commands**

Format:

```
Q
```

The command line consists of a single command name Q only. This format applies to all other commands F,W,S,M,H,C,N,E,O,V.

Function:

Same as described in Figure 6.

Example:

```
> F
```

To display Ford Drive IC in detail.

4.5/ INF Second-Level Command Group

Under "INF" second-level command, there are five third-level commands. Figure 7 shows the third-level command structure under "INF". The "INF" command group is to retrieve traffic flow information and to modify incident detection threshold values.

(1) I $M_1M_2N_1N_2N_3N_4$ Command

Format:

I M_1 M_2 N_1 N_2 N_3 N_4

The command line consists of command name I, and arguments M_1, M_2, N_1, N_2, N_3, N_4, all separated by spaces.

Function:

The "I" command is to modify the incident detection threshold values, to verify, to enter or to retrieve traffic incident information.

Where:

M_1 M_2 = The range of the mainline detector station number

N_1 N_2 N_3 N_4 = The corresponding incident detection threshold values

If only command "I M_1 M_2" is called for, where $M_1 = M_2$, then the traffic controller will verify the traffic incident information at detector station "M_1", as reported by the computer. If $M_1 \neq M_2$, and $M_2 = 0$, then the traffic controller will enter the true incident information not reported by the computer. If $M_1 < M_2$, then the traffic incident information from the detector station number M_1 to M_2 will be displayed.

Example:

```
> I 5 5 10 46 17 20
STATION 5 THRESHOLD:  10 46 17 20
> I 7 7
STATION LANE   BEGIN   END
     7    P   8:15:30   -    ---- Computer output
     7    P   8:14:00   -    ---- Traffic controller input for true
                                  incident
```

```
or  -    -       -       -    ---- Traffic controller input for
                                   incident status which cannot be
                                   verified
or  0    0       0       0    ---- Traffic controller input for false
                                   incident reported by the computer


> 1 7 0
STATION LANE   BEGIN    END
     7    P   8:14:00 8:20:00 --- Traffic controller input for true
                                  incident not detected by the
                                  computer


> I 4 11
DATE:  1978-11-23  WEATHER:  RAINY
QEW FREEWAY INCIDENT SUMMARY
STATION LANE   BEGIN     END      MAX. MAINLINE Q ON/VOL VERIFIED
     5    P   8:20:20  8:30:00          3          4 000    YES
    10    C   8:45:00  9:00:00          7          3 500    NO
```

(2) D N_1 N_2 Command

Format:

D N_1 N_2

The command line consists of command name D, and agruments N_1 and N_2, all separated by spaces.

Function:

"D" command is to retrieve 5 min detector volume and occupancy data.

Where:

N_1 = first detector number

N_2 = last detector number

Example:

```
> D 66 70
1978-11-23  7:5:0
5 min DETECTOR DATA

DETECTOR            VOLUME            OCCUPANCY %

   66                120                  19
   67                121                  20
   68                135                  25
   69                118                  14
   70                150                  28
```

(3) S N_1 N_2 Command

Format:

S N_1 N_2

The command line consists of command name S, and arguments N_1 and N_2, all separated by spaces.

Function:

The "S" command is to retrieve 30 sec mainline speed data.

Where:

N_1 = first detector station number

N_2 = last detector station number

Example:

```
S 2 4
1978-11-23   7:10:30
30 s MAINLINE SPEED DATA IN km/h
-----------------------------------------------------------------
STATION          DL            CL            PL            AVE
-----------------------------------------------------------------
   2             40            50            60             50
   3             30            40            50             40
   4             20            30            40             30
-----------------------------------------------------------------
```

(4) V N_1 N_2 Command

Format:

V N_1 N_2

The command line consists of command name V, and arguments N_1 and N_2, all separated by spaces.

Function:

The "V" command is to retrieve 5 min truck and passenger car volume count, where:

N_1 = first detector station number

N_2 = last detector station number

Example:

```
> V 6 7
1978-11-23  8:5:0
5 min TRUCK AND CAR COUNT
----------------------------------------------------------------------
STATION         TRUCK             CAR            SUM          TRUCK %
----------------------------------------------------------------------
   6              20              130            150             13
   7              20              140            160             12
----------------------------------------------------------------------
```

(5) E Command

Same as Section 4.1 (5)

4.6/ H Second-Level Command Group

Under "H" second-level command, there are 12 third-level action commands. Figure 8 shows the third-level command structure under "H". The "H" command group is to monitor and control hardware operation.

(1) BA, BN, BD Command

Format:

BA or BN or BD

These commands have the same format. The command line consists of command name BA, BN, or BD.

Function:

BA command is to enable all "BEEP" sounds

BN command is to silence individual "BEEP" sounds

BD command is to disable all "BEEP" sounds

Examples:

```
> BA
> BN
> BD
```

(2) C N_1 N_2 Command

Format:

C N_1 N_2

The command line consists of command name C, and arguments N_1, N_2, all separated by spaces.

Function:

The "C" command is to enable ramp controllers, where:

N_1 = first ramp controller identification number

N_2 = last ramp controller identification number

Example:

```
> C 3 8
```

To enable ramp controllers No. 3 to No. 8

(3) D N_1 N_2 Command

Format:

D N_1 N_2

The command line consists of command name D, and arguments N_1, N_2, all separated by spaces.

Function:

The "D" command is to enable loop detectors, where:

N_1 = first loop detector number
N_2 = last loop detector number

Example:

```
> D 43 48
```

To enable loop detectors No. 43 to No. 48.

(4) K N_1 N_2 Command

Format:

K N_1 N_2

The command line consists of command name K, and arguments N_1 and N_2, all separated by spaces.

Function:

The "K" command is to disable loop detectors, where:

N_1 = first loop detector number
N_2 = last loop detector number

```
> K 35 49
```

To disable loop detectors No. 35 to No. 49.

(5) L N_1 N_2 Command

Format:

L N_1 N_2

The command line consists of command name L, and arguments N_1, N_2, all separated by spaces.

Function:

"L" command is to enable ramp metering signal lights, where:

N_1 = first ramp metering signal light number

N_2 = last ramp metering signal light number

Example:

```
> L 2 8
```

To enable ramp metering signal lights No. 2 to No. 8

(6) M N1 N^2 Command

Format:

$M\ N_1\ N_2$

The command line consists of command name M, and arguments N_1, N_2, all separated by spaces.

Function:

The "M" command is to enable the data modem, where:

N_1 = first data modem number

N_2 = last data modem number

Example:

```
> M 1 11
```

To enable data modems No. 1 to No. 11.

(7) SE, SD Command

Format:

SE or SD

These commands have the same format. The command line consists of command name SE or SD.

Function:

SE command is to enable the changeable message sign.

SD command is to disable the changeable message sign.

Example:

```
> SE
> SD
```

(8) PN Command

__Format__;

```
P N
```

The command line consists of command name P, and arguments N separated by space.

__Function__

The "P" command is to output hardware error report, where:

N = 1, to output detector error report
N = 2, to output ramp controller error report
N = 3, to output ramp metering signal light error report
N = 4, to output data modem error report
N = 5, to output changeable message sign error report

If only command "P" is called for, then all hardware error reports will be displayed.

__Example__:

```
> P 1
DETECTORS:
57 59 61 63 65 67
> P 2
RAMP CONTROLLERS:
1 4 5
> P 3
SIGNAL LIGHTS:
4 5
> P 4
DATA MODEMS:
36 37 44
> P 5
CHANGEABLE MESSAGE SIGN:
1
```

(9) E Command

Same as Section 4.1 (5)

4.7/ JCW Second-Level Command Group

Under "JCW" second-level command, there are 13 third-level action commands. Figure 9 shows the third-level command structure under "JCW". The "JCW" command group is to monitor and control computer system operation.

(1) A N_1 N_2 N_3 Command

Format:

A N_1 N_2 N_3

The command line consists of command name "A", and arguments N_1, N_2, N_3, all separated by spaces.

Function:

The "A" command is to modify ramp controller modem address, where:

N_1 = first ramp controller identification number

N_2 = last ramp controller identification number

N_3 = ramp controller modem address for controller No. N_1 to No. N_2

Example:

```
> A 1 10 0
```

To set all 10 ramp controller modem addresses to zero.

(2) DI Command

Format:

DI

The command line consists of command name "DI".

Function:

The "DI" command is to disable the CONTROL A console interrupt mechanism.

Example:

```
> DI
CTRL A INTERRUPT DISABLED.
```

(3) EN Command

Format:

```
EN
```

The command line consists of command name "EN".

Function:

The "EN" command is to enable the CONTROL A console interrupt mechanism.

Example:

```
> EN
CRTL A INTERRUPT ENABLED
```

(4) F Command

Format:

```
F
```

The command line consists of command name F.

Function:

The "F" command is to exit from the real-time QEW software system and return control to MRDOS. The user has to carry out the following dialogue with the system.

Example:

```
F
DO YOU WISH TO STOP?  :  Y

          Files                        Records

          DATA30                          147
          DATASM                           22
          INCLOG                            2
          RAMDAT                           50
          ALMREC                           12
          XMTTAB                            2
          QEWDYN                          147
          RESDEV                            2
          ***QEW SYSTEM SHUT DOWN***

R
```

(5) **D Command**

Format:

D

The command line consists of command name D.

Function:

The "D" command is to display detector buffer content transmitted from the communication processor.

(6) **T Command**

Format:

T

The command line consists of command name T.

Function:

The "T" command is to display date, and time-of-day.

Example:

```
> T
1978-11-1  8:43:38
```

(7) M Command

Format:

M

The command line consists of command name M only.

Function:

The "M" command is to print the data modem echo test result.

(8) PN Command

Format:

PN

The command line consists of command name P and argument N, separated by space.

Function:

The "PN" command is to activate protocols 1, 2 and 3 or 4 and 5, where:

N = 1, to activate protocols 1, 2, and 3
N = 2, to activate protocols 4 and 5

If only command "P" is called for, then the protocol information between the traffic control computer and the communication processor is displayed.

(9) S N_1 N_2 N_3 Command

Format:

S N_1 N_2 N_3

The command line consists of command name "S" and arguments N_1, N_2, N_3, all separated by spaces.

Function:

The "S" command is to modify the controller modem address, where:

N_1 = First sign controller identification number

N_2 = Last sign controller identification number

N_3 = Sign controller modem address for controller number N_1 to number N_2.

Example:

```
> S 1 2 0
```

To set sign controller 1 and 2 modem addresses to zero.

(10) W Command

Format:

```
W
```

The command line consists of command name W only.

Function:

The "W" command is to modify the recording of weather conditions.

Example:

```
> W
WEATHER?     : L
PAVEMENT?    : DRY
VISIBILITY?  : CLEAR
```

(11) Y Command

Format:

```
Y
```

The command line consists of command name Y only.

Function:

The "Y" command is to display error data (cut-of-range error).

(12) Z Command

Format:

```
Z
```

The command line consists of command name Z only.

Function:

The "Z" command is to display controller command word output buffer. The output will be in terms of binary bit pattern.

(13) E Command

Same as Section 4.1 (5)

4.8/ X Second-Level Command Group

The X second-level command group is a reserved command module. It will implement new commands when they become necessary at a later date.

4.9/ System Handshake Dialogue

After the QEW command, the system will initiate the following hand-shake dialogue with the user:

```
QEW
COLD START/WARM START?
COLD
WEATHER?      :  C
PAVEMENT?     :  D
VISIBILITY?   :  C
WELCOME TO QEW FREEWAY SURVEILLANCE AND CONTROL SYSTEM PROJECT
COURTESTY OF CONTROL SYSTEMS RESEARCH - MTC
SYSTEM COLD START TIME:  1978-12-1  6:45:00
```

If "WARM" start is specified by the user, then the preceding message will become:

```
SYSTEM WARM START TIME:  1978-12-1  6:45:00
```

4.10/ System Performance Evaluation Commands

At the end of QEW control software operation, the following commands are available to evaluate QEW freeway system performance.

(1) STNOCC Command

This command will generate 5 min mainline station occupancy output on the line printer. To obtain the output, the line printer must be turned on.

(2) STNVOL Command

This command will generate 5 min mainline station volume output on the line printer.

(3) RMPOCC Command

This command will generate 5 min ramp station occupancy output on the line printer.

(4) RMPVOL Command

This command will generate 5 min ramp station volume output on the line printer.

(5) CONTOUR Command

This command will generate a traffic speed contour diagram on the 4027 Tektronix colour graphic terminal. A hardcopy is also available under this command.

(6) TT Command

This command will generate travel time diagram for various sections of the freeway on the 4027 Tektronix colour graphic terminal. A hardcopy is also available under this command.

5/ QEW FSCS OPERATION PROCEDURE

5.1/ Bringing up NOVA 4/X Control Computer System for QEW FSCS

(1) **NOVA Dasher Terminal**

- Power on Dasher terminal.
- Tap "VIEW", "ON-LINE", and "ALPHA LOCK" keys on the terminal keyboard (keys will light up).
- Make sure the terminal has paper and the "READY" light is on.

(2) **NOVA Disk**

- Power on DISK (bottom switch on left cabinet).
- Move DISK "LOAD/READY" switch to "READY" position.
- Wait about 30 s until DISK "READY" light comes on.

(3) **NOVA CPU and DASHER Terminal**

- Power on CPU (bottom switch on right cabinet).
- When DASHER Terminal Prompts with "!", enter "100033L".
- Terminal will prompt "FILENAME?". Tap "CR" key.
- MRDOS will prompt for date and time, enter date and time accordingly.
- MRDOS commands can be entered once "R" appears.

(4) **Tektronix 4027 Colour Graphic Terminal**

- Power on 4027 (switch on right front side).
- Tap the "TTY LOCK" key (it will light up).

(5) **QEW Control Software**

- Make sure communication computer HP 21MX operation software is running.
- Enter the following commands on NOVA Dasher Terminal:

```
DIR DPOF
DIR CHAN
QEW
```

5.2/ Taking Down NOVA 4/X Control Computer System

(1) Shut down QEW Control Software System

(2) Enter the following commands on NOVA Dasher Terminal:

```
RELEASE CHAN
RELEASE DPO
```

(3) Power down CPU, Dasher Terminal, Tektronix 4027

(4) Move "LOAD/READY" switch to "LOAD" position, wait for "LOAD" light to come on, power down DISK.

5.3/ Loading BDCSP Software on HP 21MX Computer

(1) Turn on floppy power with front switch.
(2) Insert floppy diskette, and close drive.
(3) Select "B" register
(4) Press "Clear Display".
(5) Set bit 0 on (Octal 1).
(6) Press "Store".
(7) Select "S" register.
(8) Press "Clear Display".
(9) Set bit 7, 8, 9, 14, 15 on (Octal 141600).
(10) Press "Store".
(11) Press "IBL", "CLEAR", and "RUN".
(12) After a minute, press "HALT".
(13) Repeat steps 3 to 12, except step 5 which should be "set bit 1 on (Octal 2)".
(14) Turn off power on floppy disk drive.

5.4/ Bringing Up BDCSP Software on HP 21MX Computer

(1) Turn key momentarily to "R" (Reset), then turn to "OPERATE".
(2) Press "HALT".
(3) Select "P" register.
(4) Press "Clear Display"
(5) Set bit 1 on (Octal 2).
(6) Press "STORE", "PRESET", "RUN".

5.5/ Bringing Up Diagnostic Software on HP 21MX Computer

(1) Repeat steps 1 to 6 in 5.4, except step 5 which should be "set bits 3,4,6,7,9,11,12 on (Octal 15330)".

5.6/ Loading QEW Software

To create the current version of QEW real-time control software, the following relocatable loader command is used.

```
RLDR/P QEW QEWBLK ^
     TOPCOM [INPEDIT, T12A, T12B, T13, T14A, T14B, T14C, T14D, T15,
       T16A, T16B, T16C, T16D, T17A, T17B, T18A, T18B, T18C]^
     COMCOM DATAIN RAMCON MODMI SLCHEK DLCHEK NXFER ^
     TRANSF [INITIALIZE, HEADING, WOOD30, UPD30S, INCDET, PERCHK, DATSTR,^
       CLDU, VTLU, RCTIME, QUOCC, ALM30S, ALM5M, AMODM, DISI, DYS1,^
       DIS2]
     GMAP [QSPD, QOCC, REQEW, CHQOR, ACCIDENT, DISMAP, RDISMAP,^
       FLASH, FACC, LOOP, IOCC, IOC2, SPD1, SPD2, RAMPS, DETECTOR, PCBOX,^
       BOX1, BOX2, BOX3] ^
       ALMSGN PCSPOL MISCO CMSCTR CORCTR ARTSIG ^
       CLOCK IBDEF ^
     FMT.LB FORT.LB
     2.0/C S/K
     QEW.LM/L
```

The execution of this command will create QEW.SV and QEW.OL files, i.e., run time control software module.

6/ FINE TUNING OF QEW FSCS SOFTWARE

The QEW control software provides the facility to fine tune the control system parameters so that a satisfactory operating condition can be achieved. This section describes the method of parameters adjustment.

6.1/ Adjustment of Ramp Control System Parameters

Depending on the changing recurrent congestion condition at the geometric bottleneck, the four-ramp control period start time frame can be changed by the operator command $TN_1N_2N_3N_4$. For example, when the recurrent congestion period is lessened, the ramp control period can be shortened. On the other hand, if the recurrent congestion period becomes longer, the ramp control period can be lengthened. The three-ramp control tables CYCLES, RAMTAB, and CYCTAB have been expanded to accommodate the three new interchanges. Command $RMN_0N_1 \ldots N_I$ shall be used to update the CYCLES table to include the new ramp controller's local cycle length in deci-second units (for example, 60 represents 6.0 s). The cycle length in the CYCLES is grouped according to the sequence of Cawthra N/B ramp, Cawthra S/B ramp, Highway 10 N/B ramp, Highway 10 S/B ramp, Mississauga N/B ramp, Mississauga S/B ramp, Southdown ramp, Winston Churchill N/B ramp, Winston Churchill S/B ramp, Ford Drive ramp.

The upstream station, local station, downstream station, and the control section assignment for each ramp will also be updated in the RAMTAB table. The detector station assignment in the RAMTAB is also grouped in the same sequence as CYCLES. The assignment in RAMTAB table will allow various co-ordinated ramp control stratagies to be implemented. The ramp control can be made responsive to one, or more than one, geometric bottleneck(s). For example, if the bottleneck is at station 19 which is between Highway 10 and Cawthra Road interchange, the upstream ramp's downstream station can be assigned 19 and the downstream ramp's upstream station can be assigned 20. In this way, the ramp control system will take the bottleneck congestion status into account. The ramp control system can also respond to the traffic incident automatically. For example, when an incident occurs, the downstream ramp will allow more vehicles to enter the mainline freeway and the upstream ramp will allow feweer vehicles to enter the freeway. The 1 min station occupancy threshold value in CYCTAB shall be adjusted to reflect the new vehicle detector signal sampling rate of 25 samples per second. For example, 1% occupancy shall be represented by 45, and at a four-lane freeway station, 1% occupancy shall be represented by 60.

6.2/ Adjustment of Ramp Queue Override Control Parameters

Due to the difference in ramp storage capacity, ramp queue detector location, and ramp demand, each ramp queue override upper-bound and lower-bound occupancy threshold value shall be adjusted by the commands $LN_1N_2..N_{10}$ and $UN_1N_2..N_{10}$. The difference in the upper-bound and lower-bound values shall be wide enough to allow for stable ramp queue override operation. The upper-bound threshold shall be set to a value corresponding to the maximum allowable queue length. The lower-bound threshold may be set to a value between 40% to 80% of the upper-bound threshold. A correlation study of the ramp queue length and ramp queue detector occupancy for each ramp is recommended for fine tuning purposes. The fine tuned threshold values can then be updated in the next version of software.

6.3/ Adjustment of Incident Detection System Parameters

Recognizing the effect of geometric difference and traffic condition variation on the incident detection system performance, each incident detection station is provided with its own set of threshold parameters which can also be adjusted by the command $IM_1M_2N_1N_2N_3N_4$.

There are four threshold parameters provided for each incident detection station. The first threshold parameter is the station occupancy threshold and is to ensure that the incident detection algorithm operates above the low-density traffic flow condition. The purpose is to reduce the incidence of unnecessary false alarms. It can be de-activated by setting it to zero. Normally, it can be set at 8% to 10%.

The second threshold parameter is the occupancy difference between the two adjacent stations. The purpose of this parameter is to ensure that the execution of the algorithm will proceed only if there is substantial occupancy difference existing between the two adjacent stations. It also helps to clear an incident condition. This parameter can be used when the first threshold parameter is set to zero. It can also be deactivated when set at zero.

The third parameter is the relative percentage occupancy difference between the two adjacent stations. This parameter is to detect the presence of a traffic flow space discontinuity due to incident. This parameter can be adjusted by observing the test value under incident condition and no incident condition. The parameter may be sensitive to geometric variation, e.g., when the station is near an on-ramp or off-ramp, or when the station is near the location where the number of lanes changes.

The fourth parameter is the relative percentage occupancy difference between two subsequent time intervals of a station. This parameter is to detect the passage of the shock wave (time discontinuity) over the detector station and is to distinquish between the incident and gradual change in traffic density due to demand change. This parameter can also be adjusted by observing the test value under incident and no incident conditions. Values of 46% for the third parameter and 17% for the fourth parameter have been found satisfactory for certain detector stations. A two- to three-month calibration period may be required when fine tuning a new system.

It should also be pointed out that there are some slight variations in the QEW incident detection algorithm when compared with the California algorithm. These variations were introduced after testing the incident detection system in the QEW FSCS. It was found that the change is required because the QEW FSCS is not operated at the optimum flow condition all the time; instead, it operates at balanced delay condition. When the system operates at the balanced delay condition, the mainline queue may still occur from time-to-time.

It is also believed that the recent improvement in the QEW corridor arterial routes and the soon-to-be-opened new Highway 403 will reduce the congestion on the QEW. There is a possibility that the QEW FSCS can then be operated at optimum flow condition. In this case, the existing incident detection algorithm may have to be adjusted or modified.

6.4/ Adjustment of Detection Hardware Monitoring Parameters

As described in Section 3, the detector hardware monitoring parameters are incorporated in the control software data section. The need for the adjustment of the detector hardware monitoring parameters may arise when there is wide variation of the traffic flow condition. Continuing monitoring of the hardware monitoring parameters and the traffic flow condition is needed to ensure satisfactory system operation.

6.5/ Generation of Changeable Message Sign Messages

Due to the demonstration nature of the QEW changeable message sign, the control software is equipped with the command capability to generate the messages, to save the messages, and to send the messages to the sign during real-time operation. The number of messages which can be generated and stored in the system is only limited by the available disk capacity. Thus, if necessary, thousands of messages can be created in the system to deal with different traffic conditions. Automatic selection and display of the message according to the traffic condition is not a requirement in the QEW demonstration system. However, this feature can be implemented when the criteria for automatic control are finalized.

6.6/ Explanation of Colour Graphic CRT Traffic Status Display

To inform the operator of the real-time traffic status, a colour graphic CRT is utilized to display the freeway and traffic in graphic symbols and colour codes. The detector station is shown in white characters with black backgound. The CMS message is shown in yellow characters with black background. The TV station is identified by a blue triangular symbol. The mainline freeway section bounded by two detector stations is coloured green if the upstream detector station occupancy is less than 20%. It is coloured yellow if the occupancy is between 20% and 30%. It is coloured red if the occupancy is greater than 30%. When the speed data are selected for the colouring option, the following colour code is used:

- greater than 100 km/h -- blue
- 80 to 100 km/h -- green

- 60 to 80 km/h -- yellow
- 40 to 60 km/h -- orange
- less than 40 km/h -- red

The interchange on-ramp is coloured according to the following convention:

- Green -- the ramp is not in queue override position.
- Yellow -- the ramp is in queue override condition, but queue detector occupancy is below upper-bound threshold value.
- Red -- the ramp is in queue override condition, but queue detector occupancy is above upper-bound threshold value.

When an incident is detected in a freeway mainline section, the colour will flash for approximately 10 s to notify the operator.

6.7/ Interface with Arterial Route Permanent Counting Station

The two permanent counting stations (PCS) on Highway 2 and Highway 5 are installed near the Credit River Bridges and can be connected to the QEW FSCS through the switched public telephone network. Through the automatic calling unit and data modem, the QEW traffic control computer can communicate with the microprocessor at the PCS location to obtain the volume and occupancy data. In addition, the two PCS can also provide hourly volume counts, truck counts, and speed spectrum count data. The two PCS are sponsored jointly by MTC and the City of Mississauga.

7/ REFERENCES

[1] Preliminary Operating and Service Manual -- Micro Controller CR710 Ramp Metering Controller. Traffic Data System.

[2] Traffic Signal Control Equipment Specifications, Traffic Engineering Office, Ontario Ministry of Transportation and Communications, Downsview, Ontario.

[3] "Development of Freeway Incident Detector Algorithms by using Pattern Recognition Techniques," TRB, Transportation Research Record 722, 1980, pp. 113 - 116.

[4] NOVA 4/X Computer System Description -- Hardware and Software Documents, Data General, 1978 Mass., U.S.A.

[5] "Specification for Cable Network Hardware and Software for QEW Expressway Traffic Control System," IS/3M, Ann Arbor, U.S.A., 1980.

[6] "QEW FSCS NOVA 4/X Real-Time Traffic Control Program Listing," Control Systems R & D, MTC, 1980.

[7] "QEW FSCS Surveillance and Control Program HIPO Software Description," Control Systems R & D, MTC, 1980.

[8] "Real-Time Programmers Guide Graphics Software for NOVA 4/X Computer," Control Systems R & D, MTC, 1981.

[9] "Traffic Control Command Language Interpreter Program Description," Control Systems R & D, MTC, 1981.

Table 1/ Maximum Controllable Cable Length

BAUD RATE (Bit/s)	48 000	24 000	12 000
Controllable cable length before modification (km)	9.6	19.2	38.4
Controllable cable length after modification (km)	32	64	128

Table 2/ Loop Centre to Loop Centre Distance

Station	Driving Lane	Centre Lane	Passing Lane
1	5.5	5.3	5.6
2	5.9	6.1	6.1
3	6.1	6.1	6.1
4	6.1	6.2	5.9
5	6.1	6.1	6.2
6	6.1	5.8, 6.1	6.1
7	6.0	6.1	6.2
8	6.1	6.1	5.9
9	8.2	8.4	8.1
10	6.1	6.1	6.1
11	6.1	6.1	6.1
12	6.4	6.4	6.4
13	6.1	6.1	6.1
14	5.5	5.5	5.5
15	4.0	3.7	3.7
16	6.1	6.1	6.1
17	5.5	5.5	5.5
18	7.0	6.1	6.1
19	6.2	6.1	6.2
20	6.1	6.3	6.4
21	5.6	6.0	5.9
22	5.8	6.0	5.9

Table 3/ CR 710 Ramp Controller Interface Assignment

8-Status Bits to Traffic Control Computer	
Bit	**Function**
0	Occupancy status
1	Cycle length status
2	Cycle length status
3	Cycle length status
4	Demand pretimed
5	Passage pretimed
6	Spare
7	Spare

8-Command Bits to CR710	
Bit	**Function**
0	Hold on line (HOL)
1	Cycle length command
2	Cycle length command
3	Cycle length command
4	Demand pretimed
5	Passage pretimed
6	0-1st message address
7	0-1st message address

Table 4/ Type 170 -- Remote Ramp Control Command Definition

Bit	Function
1	Rate Code (1)
2	Rate Code (2)
3	Rate Code (4)
4	Rate Code (8)
5	HOL
6	0
7	Command is for ramp 2, if equal to "1"
8	Command is for ramp 1, if equal to "1"

(where rate code (bit 1 to bit 4) indicates the metering rate required by the traffic control computer)

Table 5/ Operator Command Function and Overlay Subroutines

Command Groups	Functions	Subroutines
RC	To change ramp control operation	...TP12A, TP12B
QO	To modify queue occupancy override logic	...TOP13
CMS	To operate changeable message sign	...TP14A, TP14B TP14C, TP14D
G	To change graphic displays	...TOP15
INF	To retrieve traffic flow information	...TP16A, TP16B TP16C, TP16D
H	To monitor and control hardware operation	...TP17A, TP17B
JCW	To monitor and control system operation	...TP18A, TP18B, TP18C

Table 6/ Traffic Control Computer I/O Channel Usage

Channel #	Used by Routine (Task)*	For File Name
1	(QEW), (TRANSF) (GMAP)	QEW.OL
2	INITIALIZE (QEW)	DMPREC
	INITIALIZE (QEW)	XMTTAB
	HEADING (QEW)	WEATHER
3	UPD30S (TRANSF)	DATA30
	UPD30S (TRANSF)	DATA5M
	INCDET, VTLU (TRANSF)	INCLOG
	DIS1, DIS2 (TRANSF), (GMAP)	QEWDYN
	DIS2 (TRANSF)	DMPREC
	DIS2 (TRANSF)	CYCLES
	CLDU, VTLU (TRANSF)	RESDEV
5	RCTIME (TRANSF)	RAMTAB
	RCTIME (TRANSF)	CYCTAB
	QUOCC (TRANSF)	QUETAB
6	TP12B (QEW)	RAMTAB
		CYCTAB
		CYCLES
	TP14A, TP14B, TP14C (QEW)	MSGFILE
	TP14A (QEW)	PARTAB
	TP14B, TP14C, TP18C (QEW)	XMTTAB
	TP16B, TP16C, TP18C (QEW)	INCLOG
	TP16D, TP18C (QEW)	DATA30
		DATA5M
	TP18B, TP18C (QEW)	WEATHER
	TP18C (QEW)	ALMREC
	TP18C (QEW)	QEWDYN
	(GMAP)	4027 I/P
7 (RO)	QSPD, QOCC (GMAP)	QPOLY
	DETECTOR, LOOP (GMAP)	FORDPOLY
		WINPOLY
	DISMAP, RDISMAP, SPD1, SPD2 (GMAP)	ERINPOLY
	IOCC, IOC2, RAMPS, BOX1 (GMAP)	MISSPOLY
		HWYPOLY
	BOX2, BOX3 (GMAP)	CAWPOLY
	(GMAP)	4027 O/P
8	ALM30S, ALM5M, AMODM (TRANSF)	ALMREC
	ALM5M (TRANSF)	DATA5M
	DATSTR, CLDU (TRANSF)	DATAX

Table 6/ (Continued)

Channel #	Used by Routine (Task)*	For File Name
9 (RW)	FLASH, LOOP (GMAP)	TEMPFILE
10	PASSWORD, TOPCOM, HEADING (QEW) DATAIN (COMCOM), TP14D, INPEDIT, TP12A (QEW)	TTY O/P
11	PASSWORD, HEADING, TOPCOM (QEW)	TTY I/P
13	TOPCOM (QEW)	QEW.OL
14 (WO)	QSPD, QOCC (GMAP)	QPOLY
	DETECTOR, FLASH (GMAP)	FORDPOLY
		WINPOLY
	DISMAP, RDISMAP, SPD1, SPD2 (GMAP)	ERINPOLY
	IOCC, IOC2, RAMPS, BOX1 (GMAP)	MISSPOLY
		HWYPOLY
	BOX2, BOX3	CAMPOLY
15 (RO)	REQEW (GMAP)	QEWDYN

* The task to which the routine belongs or the task itself.

Table 7/ Control Software Parameters

Variables	Meaning
ALMBEEP	= 1 Sound beep at TTY.
ALMFLG	Flag word to record sign malfunction to disk file in the 30 s routine ALM30S.
AMODE	Sign stand-alone flag. = 1 in sign stand-alone mode.
APARM (6)	APARM(1): Speed threshold value APARM(2): 30 s OCC threshold value APARM(3): 5 m lower-bound OCC threshold value APARM(4): 5 m upper-bound OCC threshold value APARM(5): 5 m (15 m for exit detector) lower-bound volume threshold value APARM(6): 5 m upper-bound volume threshold value
BUFSIZ	Buffer size for sign message.
CAP (22)	Station Capacities.
CHEK1 (10)	If command transmitted ≠ command received, then CHEK1 stores command as received by NOVA from HP.
CHEK2 (10)	If command transmitted ≠ command received, then CHEK2 stores command as transmitted to HP by NOVA.
CHN1-CHN15	Channel NOS.
CNTADR (10)	Controller addresses.
CNTON (10)	Controller status flag for each ramp, on =1, off = 0.
COMDOP (8)	Control command buffer transmitted to HP.
CONMSG (2)	Control words to BDCSP.
CONTRL	Control Period, from 0 to 3 = 0 No ramps are metered = 1,3 Ramps may or may not be metered = 2 All ramps are metered at least at minimum cycle lengths.
COUNT	30 s counter.
CSTAT (10)	Controller status; 0 for O.K.; 1 not O.K.

Table 7/ (Continued)

Variables	Meaning
CTRMOD	Control period manual override by operator. 0 to 3 = control period numbers 4 = automatic advancement in control period
CYCOVR (10)	Has values ranging from 0 to 9 except 7 for each ramp. Cycle length override by operator. '1' to '5' are cycle length numbers, '0' metered green, '8' by default automatic control mode, '9' controller dropped offline, and '6' controller will select one of the valid cycle lengths based on local occupancy.
DATA30 (718)	30 s data buffer.
DETECT (545)	1 s data buffer from HP to NOVA.
DFKEQ	Error limit for CR710 traffic controllers.
DIST (20)	Distances between detector stations on the QEW section.
DNRNGE (67,3)	This contains error counts for downstream loop detectors. First 2 elements in each row contain the same information as explained in UPRNGE, except for the downstream detectors. Element 3 in each row contains error counts for double elapsed time/vehicle < 1 min.
DOCCZS (67)	Accumulated 1 s occupancy counts for downstream mainline detectors.
DTIME (67)	Accumulated downstream OCC counts for mainline detectors till volume is > 1. Then it is reset to 0.
DTIME (67)	Accumulated double elapsed time for mainline detectors till volume is > 1. Then it is reset to 0.
DTOT	No. of double loop detectors.
DVOLIS (67)	Accumulated 1 s volume counts for downstream mainline detectors.
ERHARD	Out-of-range error counts.

Table 7/ (Continued)

Variables	Meaning
ERRB5 (5)	Number of consecutive times that bit 5 of each C170 controller is on.
ERRCNT (10)	The number of consecutive error counts for different transmitted and returned commands for traffic controllers on each ramp.
EXVOL (11)	Used to accumulate 15 min exit-detector volumes.
IDATA	Number of times DATAIN routine is executed. If = 30, then NXFER subroutine is executed and TRANSF task activated. Then it is re-initialized to 0.
IFLG	Conditional flag used for overlays.
IMIN	Minimum acceptable occupancy/vehicle in pulses.
IMIN5	5 min flag.
INCDNT	When set, used to alarm the operator.
INCEX	Used to signal the execution of the incident detection program 'INCDET', 1 min after data collection has started.
INCFLAG (21)	= 0, no incident; = 1 when incident detected, appplies to all stations except the most downstream one.
INCNT (21)	Incident time counter to keep track whether the incident lasts at least 1 min.
INCREC (21)	If positive, used to indicate the record number of the incident file "INCLOG" in which the station incident information was written when the incident happened, one word for each station except those stations with no downstream stations. For example, if word 5 contains 6, it means that station 5 had an incident, the information of which was written in record number 6 of "INCLOG", so that when the station incident is cleared, some more information can be written in the same record. Again station 1 has two words, one for downstream station 2, and one for downstream station 3.

Table 7/ (Continued)

Variables	Meaning
INCSTAT (21)	= 0, no 'true' incident; = 1 after incident confirmed by 1 min persistence check (true incident).
INCTV (84)	Incident threshold values, 4 for each station, excluding the most downstream station. Station 1 has 2 sets of threshold values, one for downstream station number 2, one for downstream station number 3.
K5M	5 min data record number.
KCNT5M	5 min counter, reset every 15 min.
KCNTD	Number of consecutive times the transmitted and the received messages do not match up. Two consecutive times means sign malfunction.
KCOUNT	30 s counter for graphic program.
KDISP	Graphic file 'QEMDYN' record number.
KINC	Incident record number.
KL30S	Disk file number for DATA30.
KQCNT (10)	Time count during queue override control to reduce one cycle length number.
KSTEP (10)	Time threshold in terms of 30 s steps (e.g., 6 = 3 min) to reduce one cycle length number during queue override control.
KTHR	Time threshold in terms of 30 s steps for the ramps to have green signals during queue override control.
KTIME	Counter to activate protocol number 3.
KTLMT (10)	Time count during queue override control to meter that ramp with a green signal.
LATST	If = 1, then DATAIN will not compute speed.
LDIST (67)	Distance between the upstream and downstream loop pair.

Table 7/ (Continued)

Variables	Meaning
LIGHTS (10)	To keep track of red, yellow and green bits on 30 s basis.
LTCOND (10)	To keep track of which lights (red, amber or green) were on, updated every second for 30 s.
LTSTAT (10)	Light status, 1 not O.K.
LVCIS (67)	Long vehicle lane counts per second.
MAXOCC	Maximum number of pulses (occupancy) per second.
MAXCYC	Maximum number of cycle lengths chosen in a controller.
MAXSPD	Maximum speed allowed.
MAXVEH	Maximum acceptable volume counts per second.
MCNT (10)	Index to map ramp numbers to corresponding positions in command (transmit or return) buffer.
MDMIET (73)	The lowest order digit of i^{th} word contains number of detectors associated with the i^{th} address, and the high order digits point to the starting location in XIB (180) which again points to a location in XREFTB vector, where the first detect ID is stored.
MDY	Month, Date and Year.
MSEID	Message sequence number for protocol number 1.
METER (10)	= 0, no ramp metering, = 1, ramp metering for that ramp.
MODCYC (10)	Indicates which control mode each ramp is in, whether it be under local occupancy, upstream occupancy or upstream volume control, or a combination of the three above, or queue override control, or manual override, or no control.
MODEM (148)	Modem status (0 = good, > 0 no good)
MODEM1 (73)	Accumulated modem response error counts, done every second.

Table 7/ (Continued)

Variables	Meaning
MODEMT (299)	MODEMT (4 - 151) modem response from HP; MODEMT (151 - 299) total accumulated modem response error counts.
MODNO (73)	Contain modem numbers for corresponding modem addresses.
MTLSGN (2)	Contains destination addresses for sign message.
NALREC	'ALMREC' record number.
NBRAMP	NBRAMP = 1 => transmitting commands to C170 N/B ramps; NBRAMP = 0 => transmitting commands to C170 S/B ramps, and comparing transmitted commands to return messages for the C170 controllers, each of which controls 2 ramps. NBRAMP takes on the value of 0 and 1 every other second.
NDD	Number of demand detectors.
NED	Number of exit detectors.
NEPROR (73)	Response error recovery counter for all modem addresses.
NMSGRX	Number of messages received via protocol number 5.
NMSGTX	Number of messages transmitted.
NOBEEP	= 1 Silence beep permanently; = 0 initially.
NPD	Number of passage detectors.
NQD	Number of queue detectors.
NRAMPS	Number of ramps.
NSTNS	Number of stations.
NUMMDM	Total number of modem addresses in the system.
NWORDS	Number of words in the transmitted message including the address and control characters.
NWREC	Number of weather records.

Table 7/ (Continued)

Variables	Meaning
OCCDBI (21)	Downstream station occupancy at the instant of incident occurence.
OCC1S (46)	Accumulated 1 s occupancy counts for single loop detectors.
P4FLG	Flag word to initiate protocols #4 and #5; = 1, in the process of initiating P#4, > 1 in the process of initiating P#5, = 0, no message being transmitted, or no initiation of either protocol.
PASSOK	= 0, CMS Commands only; = 1, entire operator commands.
PR1ERR	Error counter for protocol #1.
PR2ERR	Error counter for protocol #2.
PR3ERR	Error counter for protocol #3.
PR4ERR	Error counter for protocol #4.
PR5ERR	Error counter for protocol #5.
PROTOCOL	Protocol number.
PSPRT	Time in seconds before protocol #3 is executed.
QIND (10)	= 1 ramp under queue override control; = 0, the opposite.
QLNTH (10)	Queue length for each ramp.
QOCC (13)	Smoothed queue detector occupancies (pulses).
QOCCLB (10)	Lower-bound occupancy threshold values for queue override control (pulses).
QOCCUB (10)	Upper-bound occupancy threshold values for queue override control (pulses).
OCCUBI (21)	Upstream station occupancy at the instant of incident occurence.
QOVR (10)	= 0 by default, queue override allowed for that ramp, = 1 no queue override for that ramp.

Table 7/ (Continued)

Variables	Meaning
QWAIT (10)	Queue waiting time for each ramp.
RAMCYC (10)	Cycle length ID for each ramp ranging from 0 to 6 transmitted to the controller.
RCV170 (5)	Contains return messages from C170 controllers.
RCVBUF (45)	45-word buffer for received message.
REPLY (2)	Response from BDCSP to control words from NOVA.
S1, S2	Smoothing constants for speed computation.
SGNADR (2)	Sign addresses for two different signs.
SGNSTAT (2)	Sign status words for two signs: = 0, O.K.; > 0, not O.K.
STDLN	Maximum length of a short vehicle.
STIMI (46)	Accumulated occupancy counts for single-loop detectors till volume is > 1. Then it is reset to 0.
STOT	Number of single-loop detectors.
THRCP (4)	Time thresholds for automatic advancement in control periods 1, 2, 3, and 0.
THRMQL	Station occupancy threshold for calculating the end of mainline queue.
TIMER	Word in common block KLOCK to wake up COMCOM.
TSKO	Used to wake up Task ALMSGN.
TSKI	Used to wake up Task TRANSF.
TSKST1	= 0 Task TRANSF suspended; = 1 in execution.
TSKST2	= 0 Task ALMSGN suspended; = 1 in execution.
TT (20)	Travel times between stations on QEW.
UOCCIS (67)	Accumulated 1 s occupancy counts for upstream mainline detectors.

Table 7/ (Continued)

Variables	Meaning
UPRNGE (113,2)	In each row, first element contains error counts for volume > MAXVEH, second element contains error counts for occupancy/vehicle < IMIN. First 46 rows are for single loop detectors and the remaining 67 rows are for upstream mainline loop detectors.
UPTIM (67)	Accumulated upstream occupancy counts for mainline detectors till volume > 1, then it is reset to 0.
UVOLIS (67)	Accumulated 1 s volume counts for upstream mainline detectors.
UWAIT (67)	Used to count up to 2 s when certain calculations are performed under certain conditions.
VOLIS (46)	Accumulated 1 s volume counts for single-loop detectors.
VSPEED (67)	Lane speed smoothed every second.
WARM	= 1, warm start; = 0, cold start
WEATHER (33)	First three words hour, minute, second; the rest are weather conditions.
XMT170 (5)	Contains commands transmitted to C170 controllers.
XMTBDF (45)	45-word buffer for transmitted message.
XREF (113)	Indexes to process corresponding detector data in DATA30.
XREFTB (180)	Contains detector numbers.
XTB (180)	Pointers to XREFTB vector to access the required detector IDs for each modem connected with detectors.

Table 7/ (Continued)

DATSTR

Variables	Meaning
DATAX	Array used to pass data between "DATSTR" data gathering routine and "CLDU" routine -- array is used to write to temporary disk file at same name. File is read by "CLDU".
DNOCC	Downstream lane occupancy data index for DATA30 file records.
DNSPO	Downstream lane speed data index for DATA30 file records.
DNVOL	Downstream lane volume data index for DATA30 file records.
DT30S	Array which receives data from logical record of DATA30 file.
INCPRG	Array which signals incident in progress -- this flag is used to discriminate between new incidents and incidents which have already occurred but are not yet cleared. 0 = no incident in progress 1 = incident in progress
INCSTAT	Array which signals incident detected -- does not differentiate between new incidents and incidents still in progress. 0 = no incident 1 = incident detected
KL30S	Starting physical record index for logical records of "DATA30" file -- used to read and write to DATA30 -- set [1st physical record of current "DATA30" logical record] + 12 (starting record for next time slice)
REC30	Used to read "DATA30" disk file.
STN	Current collecting station being checked. Counter loop index.
T	Time slice index -- counts number of time slices taken.
TREC	Starting physical record of current "DATAX" logical record.

Table 7/ (Continued)

CLDU

Variables	Meaning
CL	Element of (lane: occ., vol., or spd.) array containing data for the centre lane at the station upstream from the incident.
DATAX	Array accepts data from "DATAX" disk file.
DL	Upstream station driving lane data index.
DNCL	Downstream station centre lane data index.
DNDL	Downstream station driving lane data index.
DNPL	Downstream station passing lane data index.
HMS	hour, minute, second -- common block array containing current system time.
INCPRG	Same as in DATSTR routine.
INCOT	Incident occupancy (occ.) threshold.
INCST	Incident speed (spd.) threshold.
INCVT	Incident volume (vol.) threshold.
INCRMT	Value by which upstream station indices are incremented to calculate downstream station indices.
INCSTAT	Same as in DATSTR routine.
LNOCC	Array containing lane occupancy data for time slice.
LNSPD	Array containing lane speed data for time slice.
LNVOL	Array containing lane volume data for time slice.
MIN	(lane occ., vol., spd.) array index of lane with minimum value.
RDRCD	[physical, logical] record of "RESDEV" disk file to be written to.

Table 7/ (Continued)

CLDU (Continued)

Variables	Meaning
RESDEV	Array containing data to be written to disk file "RESDEV". Used by CLDU routine to pass 'lane of incident' selections to VTLU routine for vote counting.
RSDVR	Array giving record of "RESDEV" contain data for a given station -- similar to 'INCREC' array used earlier with INCLOG.
SIGMA	Value derived through a comparison between the lane with minimum value and the station average. This is tested against the corresponding incident threshold value to determine if the difference is great enough to base a decision on.
STN	Main DO-LOOP index -- indicates current station.
T	Same as in DATSTR.
TREC	Same as in DATSTR.
TOCC	Occupancy index for RESDEV array -- used to store occ. CLDU selection in array to be passed by disk to VTLU.
TSPD	Speed index for RESDEV array.
TVOL	Volume index for RESDEV array.

Table 7/ (Continued)

VTLU

Variables	Meaning
CNTR	Number of votes for centre lane incident.
DRIV	Number of votes for driving lane incident.
HMS	h, min, s -- array containing current system time.
INCLAN	Lane of incident.
INCLOG	Array used to hold "INCLOG" disk file data while lane at incident decision is added.
INCPRG	Same as in DATSTR routine.
INCSTAT	Same as in DATSTR routine.
MDY	Month, day, year -- array containing current system date.
OCCMAJ	Decision of occupancy VTLU.
OCCVOT	Array extracted from "RESDEV" file - contains three votes from occupancy CLDU.
OCWT	Weighting given to lane selection based on occupancy in final vote taking.
PASS	Number of votes for passing lane incident.
RESDV	Local array used to handle "RESDEV" file data. Data read from disk - utilized - written back to disk.
SPDMAJ	'Lane of incident' decision of speed VTLU.
SPDVOT	Array containing votes from speed CLDU.
SPWT	Weighting given to 'SPDMAJ' in final vote taking.
VOLMAJ	'Lane of incident' decision of volume VTLU.
VOLVOT	Array containing votes from volume CLDU.
VLWT	Weighting given to 'VOLMAJ' in final vote taking.

Table 7/ (Continued)

Variables	Meaning
ALXQCT	ALMSGN task execution count.
CHQ64	Channel for "QTY:64".
CHQ01	Channel for "QTY:01".
RBFUB	Receive buffer RBUFF length in bytes.
RBUFF (25)	Receive buffer.
TBFUB	Transmit buffer TBUFF length in bytes.
TBUFF (5)	Transmit buffer.
TSK3	Message key in common for task "ALMSGN".

Table 8/ QEW Data File Structure

File Name	Logical Record Length	Physical Record Length	No. of Physical Records Per Logical Records
DATA30	718 words (768)*	128 bytes	12
DATA5M	430 words (448)*	128 bytes	7
QEWDYN	457 words (512)*	128 bytes	8
INCLOG	25 words	50 bytes	1
ALMREC	14 words	28 bytes	1
XMTTAB	45 words	90 bytes	1
RESDEV	26 words	52 bytes	1
WEATHER	36 words	72 bytes	1

* Buffer length

Table 9/Ramp Metering Supports File Structure

File Name	Logical Record Length	Physical Record Length	No. of Physical Records Per Logical Records
RAMTAB	40 words	80 bytes	1
CYCTAB	25 words	50 bytes	1
QUETAB	256 words	128 bytes	4
CYCLES	50 words	100 bytes	1

MRDOS ——— QEW -- To activate on-line real-time
traffic control system

Figure 1/ First-Level Command Structure

QEW

- RC -- To change ramp control operation
- QO -- To modify queue occupancy override logic
- CMS -- To operate changeable message sign
- G -- To change graphic displays
- INF -- To retrieve traffic flow information
- H -- To monitor and control hardware operation
- JCW -- To monitor and control system operation
- X -- Reserved for new commands

Figure 2/ Second-Level Command Structure Under QEW

RC

- C $N_1N_2..N_iM\#$ - To override ramp controller metering rate
- P_N - To change advancement of ramp control period
- T $N_1N_2N_3N_4$ - To change ramp control period start time frame
- RM$N_0N_1N_2..N_i$ - To update co-ordinated ramp control tables "RAMTAB", "CYCTAB", "CYCLES"
- E - To exit command

Figure 3/ Third-Level Command Structure Under "RC"

QO	- N#	- To enable queue override logic on ramp No. #
	- D#	- To disable queue override logic on ramp No. #
	- L N_1 N_2...N_{10}	- To change lower-bound Q-OCC threshold values on all 10 ramps in terms of number of pulses
	- U N_1 N_2...N_{10}	- To change upper-bound Q-OCC threshold values on all 10 ramps in terms of number of pulses
	- T M S N_1 N_2...N_{10}	- To set queue override time limit, M, before steady green and number of 30-s time step, N_i, i = 1, 2, .., 10 during cycle length reduction process
	- P	- To print out common buffer related to ramp metering strategy and queue override
	- E	- To exit command

Figure 4/ Third-Level Command Structure Under "QO"

CMS	- A	- To add a message to the message file
	- B	- To print message transmitting CMS buffer and receiving buffer
	- D#	- To delete a message with message number # from the message file
	- L	- To list all transmitted CMS messages
	- S N_1 $N_2 \ldots N_i$ M #	- To send a message with message number # from message file to sign CMS controller with identification number N_1 $N_2 \ldots N_i$
	- P#	- To print a message with message number # from message file
	- E	- To exit command

Figure 5/ Third-Level Command Structure Under "CMS"

G

- Q -- Main QEW display
- F -- Ford Drive Interchange (IC) display
- W -- Winston Churchill IC display
- S -- Southdown Road IC display
- M -- Mississauga Road IC display
- H -- Highway 10 IC display
- C -- Cawthra Road IC display
- N -- To freeze display for hard copies
- E -- To exit command
- O -- To use occupancy for colour display
- V -- To use speed for colour display

Figure 6/ Third-Level Command Structure Under "G"

INF

- I $M_1M_2N_1N_2N_3N_4$ - To modify incident detection threshold and to retrieve incident information
- D N_1N_2 - To retrieve 5-min detector data
- S N_1N_2 - To retrieve 30-s mainline speed data
- V N_1N_2 - To retrieve 5-min truck and passenger car volume count
- E - To exit command

Figure 7/ Third-Level Command Structure Under "INF"

H

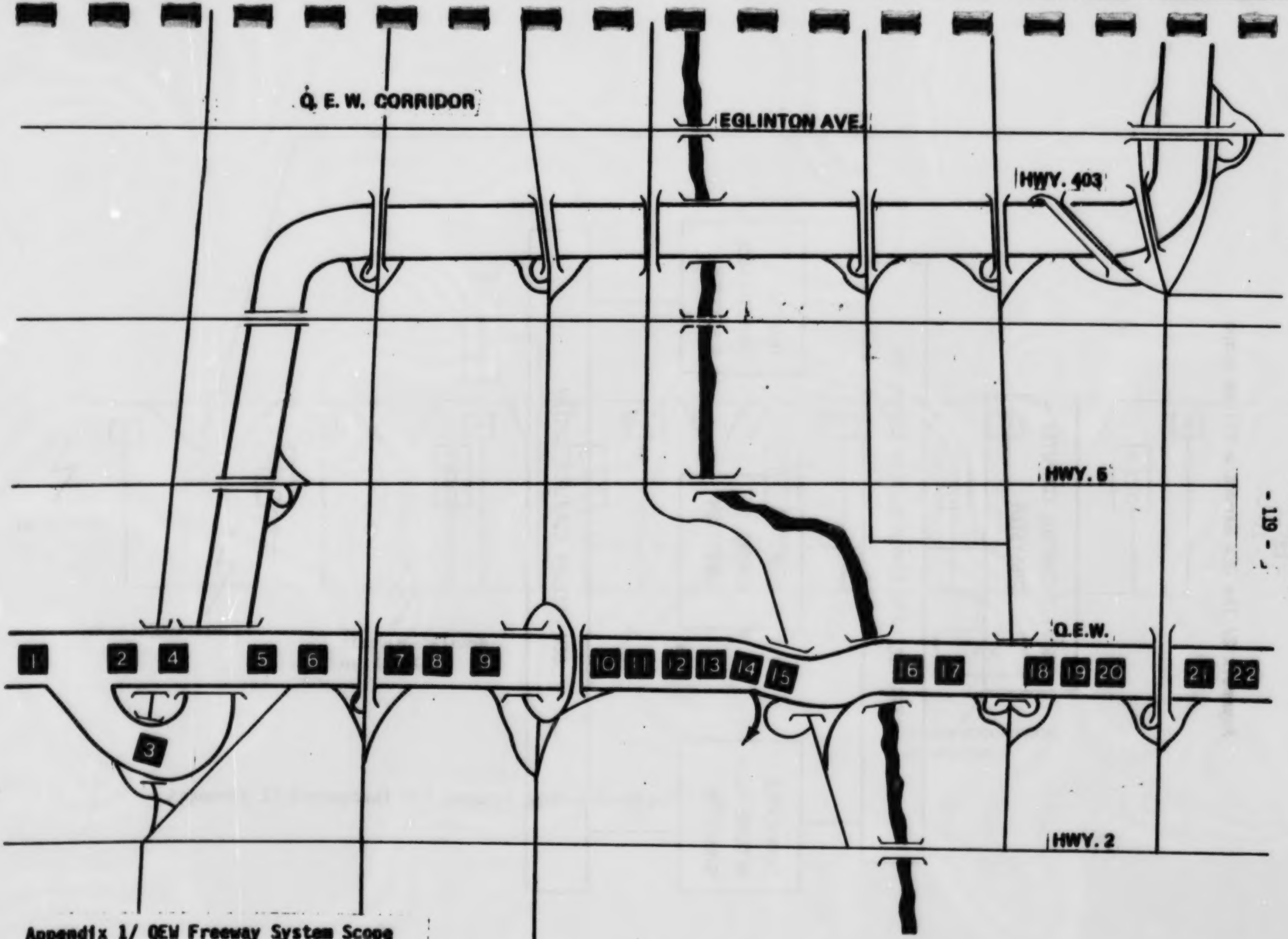
- BA - To enable all "BEEP" sound
- BN - To silence individual "BEEP" sound
- BD - To disable all "BEEP" sound
- C N_1 N_2 - To enable ramp controllers
- D N_1 N_2 - To enable loop detectors
- K N_1 N_2 - To disable loop detector
- L N_1 N_2 - To enable ramp metering signal lights
- M N_1 N_2 - To enable data modems
- SE - To enable changeable message sign
- SD - To disable changeable message sign
- PN - To print hardware error report
- E - To exit command

Figure 8/ Third-Level Command Structure Under "H"

JCW

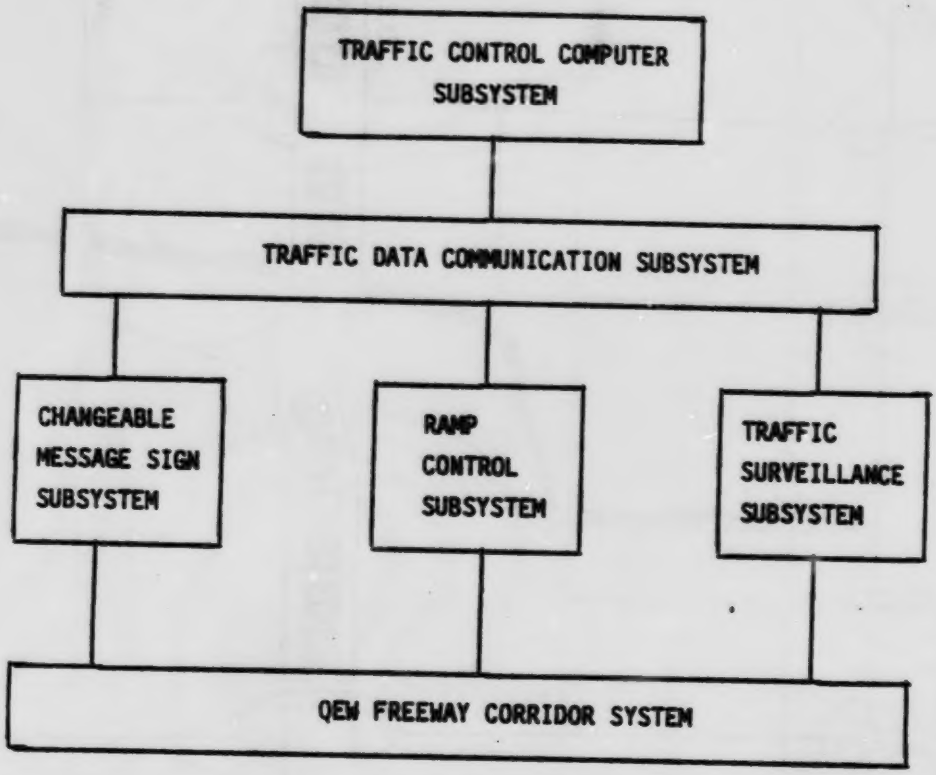
- A N_1 N_2 N_3 - To modify ramp controller modem address
- DI - To disable console interrupt mechanism
- EN - To enable console interrupt mechanism
- F - To exit from QEW system
- D - To display detector buffer content transmitted from communication processor
- T - To display date and time of day
- M - To display data modem echo test result
- PN - To control and display computer-to-computer communication information
- S N_1 N_2 N_3 - To modify sign controller modem address
- W - To modify weather conditions
- Y - To display detector out-of-range error data
- Z - To display controller command word output buffer
- E - To exit command

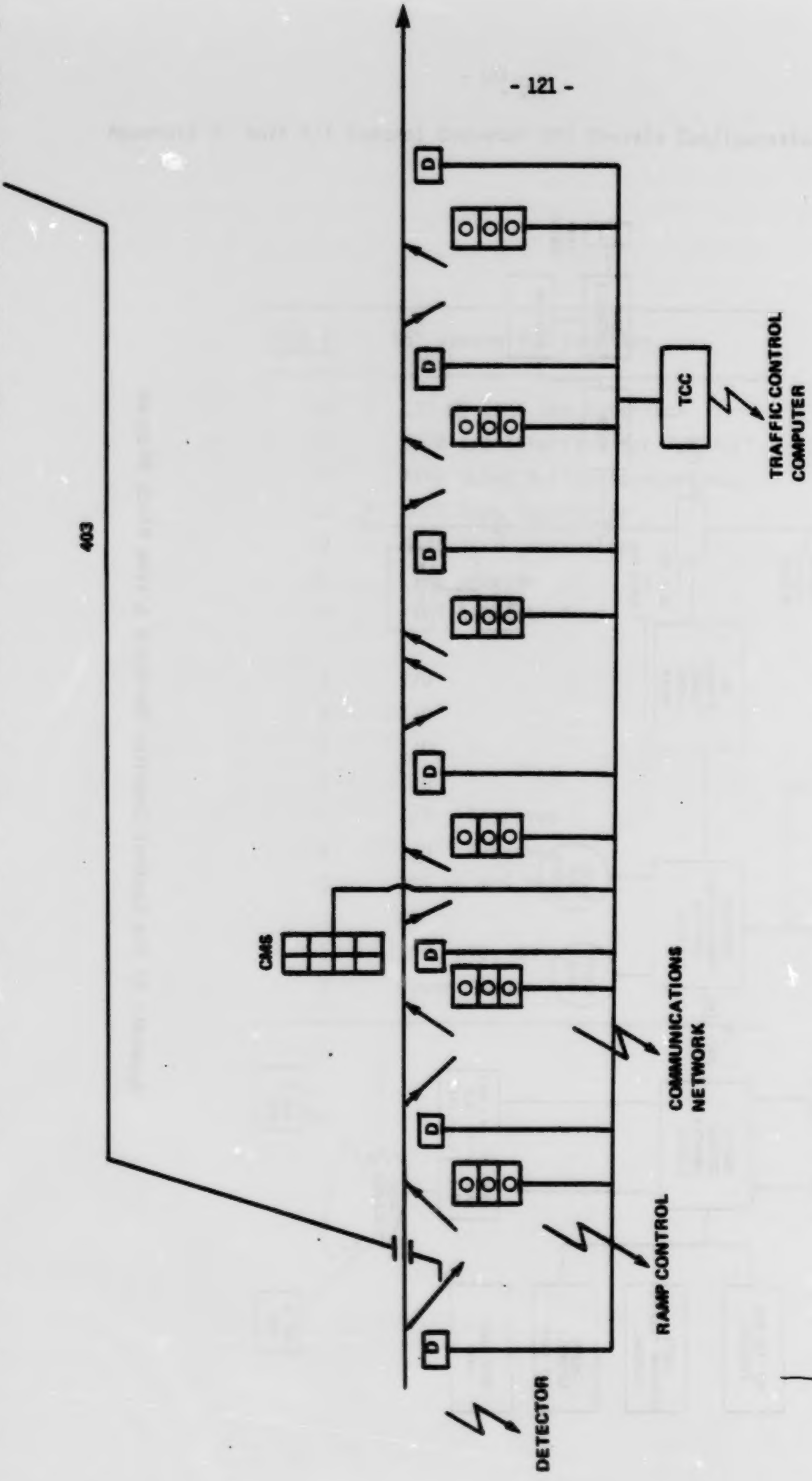
Figure 9/ Third-Level Command Structure under "JCW"


Q. E. W. CORRIDOR
EGLINTON AVE.
HWY. 403
HWY. 5
Q.E.W.
HWY. 2
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

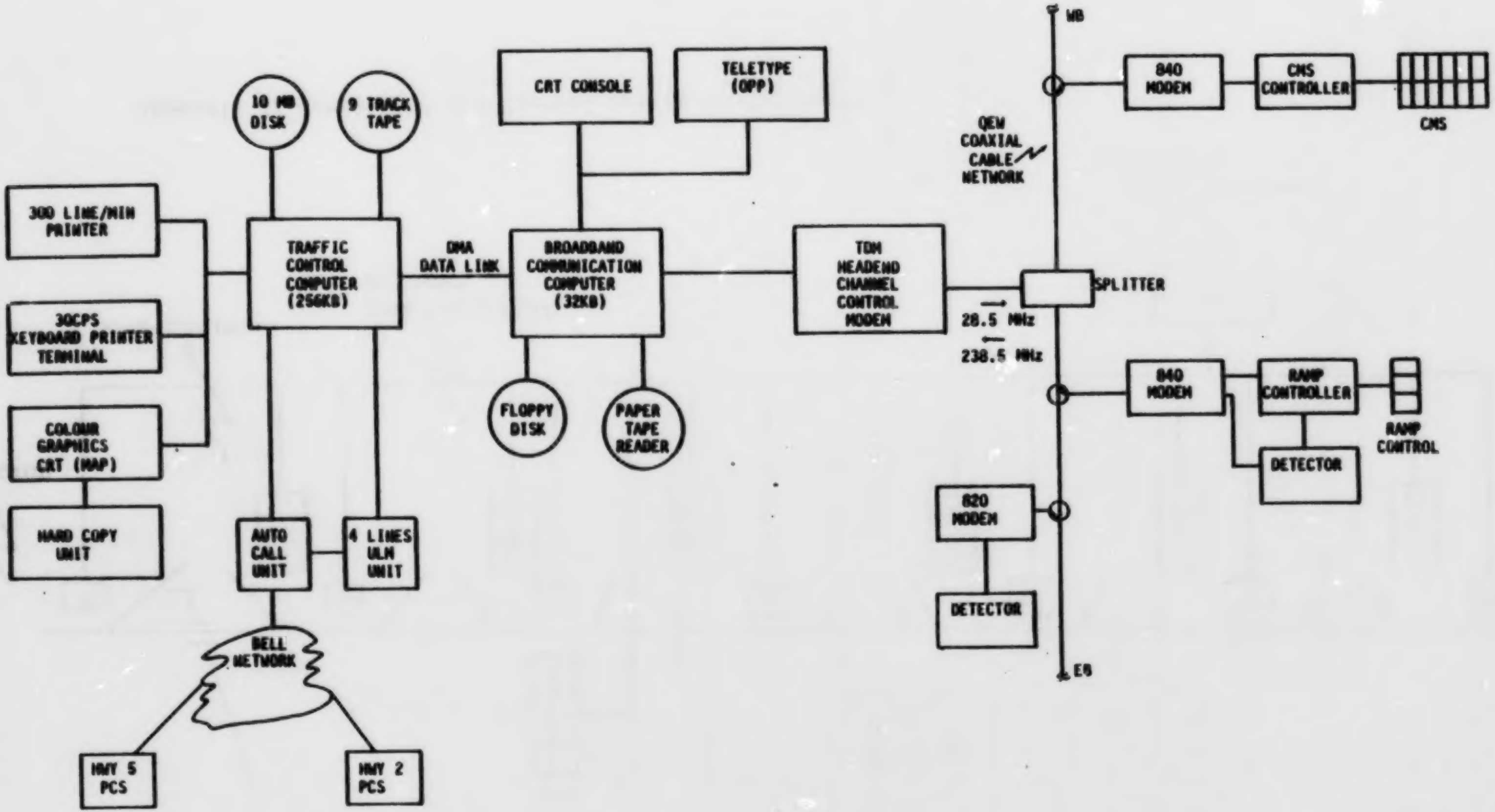
Appendix 1/ QEW Freeway System Scope

Appendix 2/ The QEW Hardware System Scope


TRAFFIC CONTROL COMPUTER SUBSYSTEM
TRAFFIC DATA COMMUNICATION SUBSYSTEM
CHANGEABLE MESSAGE SIGN SUBSYSTEM
RAMP CONTROL SUBSYSTEM
TRAFFIC SURVEILLANCE SUBSYSTEM
QEW FREEWAY CORRIDOR SYSTEM


403
CMS
D
TCC
TRAFFIC CONTROL COMPUTER
COMMUNICATIONS NETWORK
RAMP CONTROL
DETECTOR

Appendix 3/ Conceptual QEW Control System Configuration


10 MB DISK
9 TRACK TAPE
CRT CONSOLE
TELETYPE (OPP)
WB
840 MODEM
CMS CONTROLLER
CMS
QEW COAXIAL CABLE NETWORK
300 LINE/MIN PRINTER
TRAFFIC CONTROL COMPUTER (256KB)
DMA DATA LINK
BROADBAND COMMUNICATION COMPUTER (32KB)
TDM HEADEND CHANNEL CONTROL MODEM
SPLITTER
28.5 MHz
238.5 MHz
30CPS KEYBOARD PRINTER TERMINAL
840 MODEM
RAMP CONTROLLER
RAMP CONTROL
DETECTOR
COLOUR GRAPHICS CRT (MAP)
FLOPPY DISK
PAPER TAPE READER
820 MODEM
HARD COPY UNIT
AUTO CALL UNIT
4 LINES ULN UNIT
DETECTOR
EB
BELL NETWORK
HWY 5 PCS
HWY 2 PCS

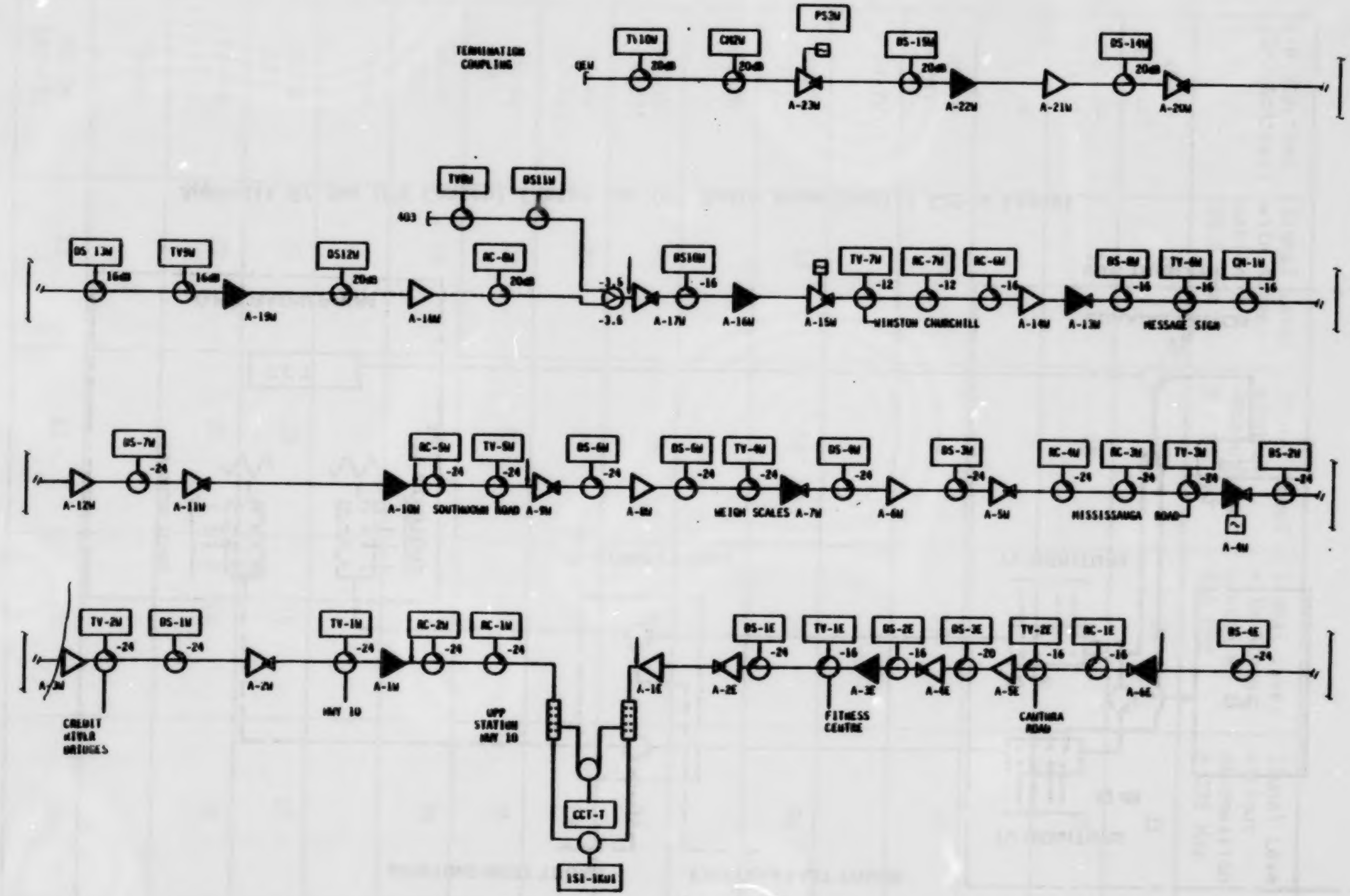
Appendix 4/ The Control Computer Hardware System Block Diagram

Appendix 5/ NOVA 4/X Control Computer CPU Chassis Configuration

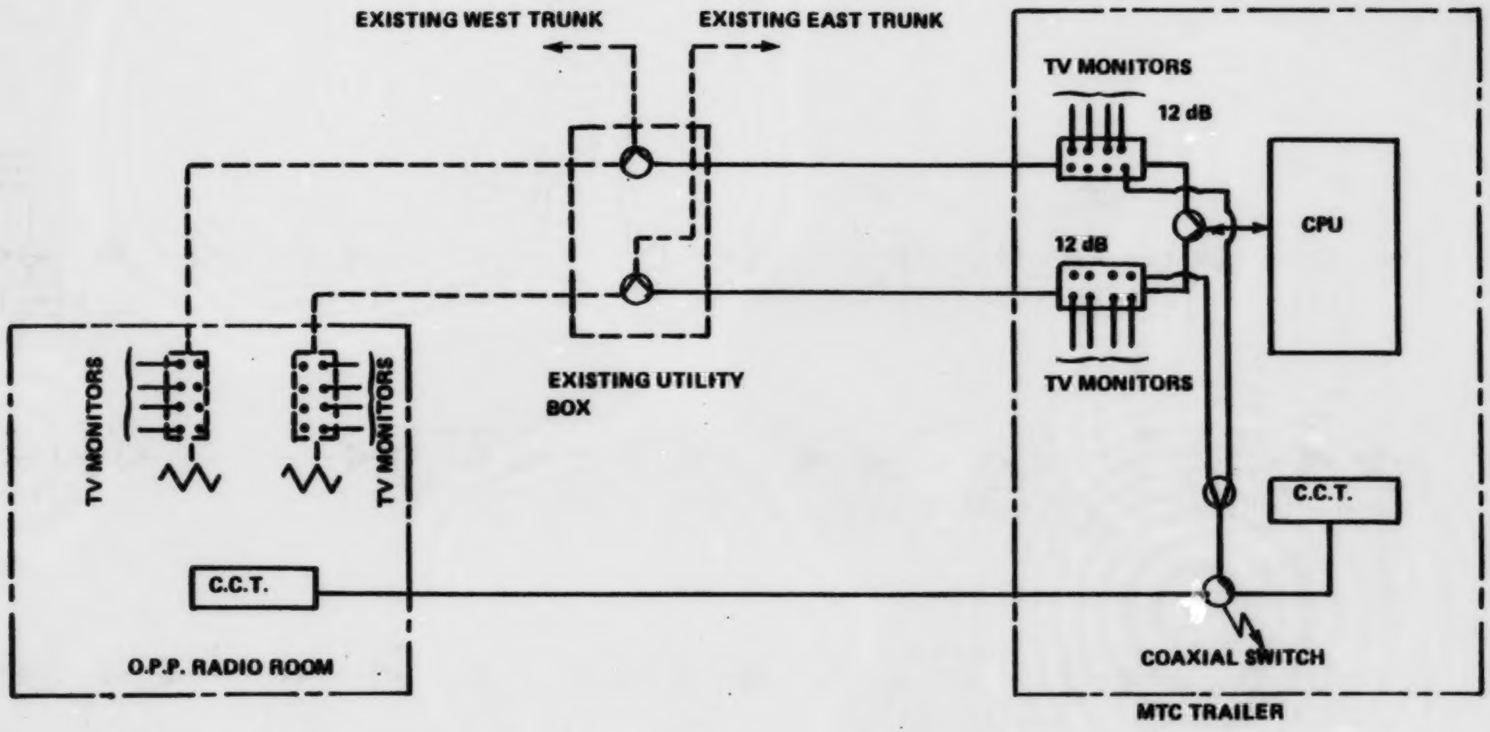
Slot #	15" square PCB Function
16	ISI HP 21MX DMA Interface
15	4075 I/O Interface for Tek 4027
14	4241 ULM-5 4-line Asynchronous
13	6027 Tape Controller
12	6045 Disk Controller
11	Line printer
10	4014 I/O Interface
9	I/O
8	I/O
7	I/O
6	I/O
5	I/O Reserved
4	I/O Reserved
3	256 kB MOS Memory
2	FPU
1	CPU, RTC, MMPU,
0	Power Supply

Appendix 6/ HP 21MX Communication Computer CPU I/O Slot Assignment

Slot #	Device
6	Computer-computer channel DMA control
10	Head-end receive channel
11	Teletype
12	Computer-computer channel data port
13	Paper tape reader
14	Time base generator
15	Head-end transmit channel


TERMINATION COUPLING
QEW
TV10W
20dB
CN2W
20dB
PS3W
A-23W
DS-15W
20dB
A-22W
A-21W
DS-14W
20dB
A-20W
403
TV8W
DS11W
DS 13W
16dB
TV9W
16dB
A-19W
DS12W
20dB
A-18W
RC-8W
20dB
-3.5
-3.5
A-17W
DS10W
-16
A-16W
A-15W
TV-7W
-12
RC-7W
-12
RC-6W
-16
WINSTON CHURCHILL
A-14W
A-13W
DS-8W
-16
TV-6W
-16
CM-1W
-16
MESSAGE SIGN
A-12W
DS-7W
-24
A-11W
A-10W
RC-5W
-24
TV-5W
-24
SOUTHDOWN ROAD
A-9W
DS-6W
-24
A-8W
DS-5W
-24
TV-4W
-24
WEIGH SCALES
A-7W
DS-4W
-24
A-6W
DS-3W
-24
A-5W
RC-4W
-24
RC-3W
-24
TV-3W
-24
MISSISSAUGA ROAD
A-4W
DS-2W
-24
A-3W
TV-2W
-24
CREDIT RIVER BRIDGES
DS-1W
-24
A-2W
TV-1W
-24
HWY 10
A-1W
RC-2W
-24
RC-1W
-24
OPP STATION HWY 10
CCT-T
ISI-IKUI
A-1E
A-2E
DS-1E
-24
TV-1E
-16
FITNESS CENTRE
A-3E
DS-2E
-16
A-4E
DS-3E
-20
A-5E
TV-2E
-16
CAWTHRA ROAD
RC-1E
-16
A-6E
DS-4E
-24

Appendix 7/ QEW Broadband Coaxial Cable Network Layout


EXISTING WEST TRUNK
EXISTING EAST TRUNK
TV MONITORS
12 dB
CPU
12 dB
TV MONITORS
EXISTING UTILITY BOX
TV MONITORS
TV MONITORS
C.C.T.
C.C.T.
O.P.P. RADIO ROOM
COAXIAL SWITCH
MTC TRAILER

Appendix 8/ New QEW Control Center and OPP Radio Room Coaxial Cable Layout

Appendix 9/ Coaxial Cable Network Tap Station Carrier Level

Tap Station Carrier Levels

Tap Stations (CPU Address)	Signal Level with Attenuation @ 28.8 MHz	Signal Level without Attenuation @ 28.8 MHz	Signal Level with Attenuation @ 238 MHz	Signal Level without Attenuation @ 238 MHz
RC-1W (10-13)	20	30	20	33
RC-2W (14-17)	18	28	15	25
TV-1W				
DS-1W (20)	20	33	19	47
TV-2W				
DS-2W (21)	18	48	17	56
TV-3W				
RC-3W (24-27)	20	32	20	44
RC-4W (30-33)	22	28	17	29
DS-3W (34)	22	32	16	35
DS-4W (35)	18	24	17	38
TV-4W				
DS-5W (36)	23	29	15	27
DS-6W (37)	22	22	18	33
TV-5W				
RC-5W (40-43)	21	21	20	20

Appendix 10/Broadband Amplifier Input and Output Level

System Levels as set November 10, 1980

Amplifier
Type

Amplifier STN	A-1W	A-2W	A-3W	A-4W	A-5W	A-6W	A-7W	A-8W	A-9W	A-10W	A-11W	A-12W
CH-3V 61.25MHz	+20 / +30	+20 / +30	+16 / +30	+16 / +25	+12 / +26	+12 / +24	+20 / +24	+8 / +28	+8 / +28	- / -	- / -	- / -
CH-4V 67.25MHz	+20 / +30	+20 / +30	+16 / +30	+16 / +25	+14 / +28	+14 / +24	+20 / +24	+8 / +28	+8 / +28	+15 / +22	+15 / +22	+13 / +25
CH-6V 83.25MHz	+26 / +38	+26 / +40	+20 / +40	+25 / +30	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -
CH-7V 175.25MHz	+6 / +24	+6 / +28	+6 / +28	+10 / +25	+6 / +24	+6 / +24	+8 / +28	- / -	- / -	- / -	- / -	- / -
CH-10V 193.25MHz	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -
CH-12V 205.25MHz	+12 / +30	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -	- / -
CH-13V 211.25MHz	+10 / +30	+10 / +30	+10 / +32	+12 / +30	+10 / +30	+8 / +28	+10 / +30	+6 / +30	+6 / +30	+13 / +26	+13 / +26	+7 / +28
Test C.W. 245MHz	+10 / +30	+12 / +33	+10 / +33	+12 / +25	+10 / +33	+10 / +32	+12 / +32	+8 / +32	- / -	- /	- /	- /
FSK 23.5 MHz	+33 / +33	+21 / +50	+39 / +39	+30 / +39	+28 / +34	+39 / +39	+30 / +45	+39 / +39	+28 / +39	+33 / +33	+28 / +35	+29 / +29
ISI 28.8 MHz	+54 / +54	+45 / +62	+51 / +51	+45 / +62	+56 / +68	+56 / +56	+50 / +65	+59 / +59	+48 / +50	+48 / +48	+42 / +50	+44 / +44

Appendix 10/ Cont'd

SYSTEM CARRIER LEVELS dBmV

I/P - O/P

Coil freq.	Band		A-18W	A-19W	A-20W	A-21W	A-22W	A-23W
30 MHz	Sub Band		- -	- -	+21 +35	- -	- -	+20 +35
88 MHz	Low Band		+10 +21	+10 +18	+10 +20	+10 +19	+10 +17	+10 +19
174 MHz	Mid Band		+10 +24	+10 +20	+10 +23	+10 23	+10 +20	+10 +23
216 MHz	High Band		+10 +27	+10 +22	+10 +24	+10 +25	+10 +22	+10 +25
300 MHz	Super Band	- -	+10 +30	+10 +24	+10 +27	+10 +28	+10 +24	+10 +28

Appendix 11/ The Connection Between Data Modem and The Coaxial Cable Tap Station

Tap Station Identification	Modem Address	Traffic Station Identification
W/B		
RC1W	10 - 13	C1
RC2W	14 - 17	C2, S18
DS1W	20	S17
DS2W	21	S16
RC3W	24 - 27	C3
RC4W	30 - 33	C4, S15
DS3W	34	S14
DS4W	35	S13
DS5W	36	S12
DS6W	37	S11
RC5W	40 - 43	C5, S10
DS7W	45	S9
CM1W	44	CMS1
DS8W	46	S8
RC6W	50 - 53	C6, S7
RC7W	-	-
DS10W	54	S6
DS11W	-	-
RC8W	60 -63	C7, S5
DS12W	64, 65, 66	S4, S3
DS13W	70	S2
DS14W	71	S1
DS15W	-	-
CM2W	-	-
E/B		
DS1E	100	S19
DS2E	101	S20
DS3E	102	off-ramp
RC1E	104 - 107	C8, S21
DS4E	110	S22

Appendix 12/ Modem Address Assignment

The table below shows the assignment of modem addresses to traffic devices. Addresses with no devices attached are followed by ---. Detector inputs are labeled as single detectors (SD) or double detectors (DD). Note that single detectors and double detectors cannot be mixed on a single modem address.

The modem # is followed by (1) or (4) to indicate whether the modem is a 1-address or 4-address modem. New modems in the system expansion are marked with *. As indicated below, one address must be unused between West and East detectors, to allow for signal propagation delays.

The reset address is critical for system timing. Reset after 148 addresses (reset address set to 144) gives almost exactly 25 scans per second [((48 000 bits per second/13 bits per word)/148 words per scan) = (24.95 scans per second)]

Modem # (#adrs)	Octal Modem Address	Traffic Device #	# Inputs	# Outputs	Approx ft W or E of Head-end
	0-7	---(reserved for system timing)			
1 (4)	10	C1	8	8	164 W
	11	C1	8 (status)		
	12	D1,D2,D5,D7	4 SD		
	13	---			
2 (4)	14	C2	8	8	302 W
	15	C2	8 (stat)		
	16	D3,D4,D6,D8,D9	5 SD		
	17	D10,D11,D12,D13,D14,D15	6 DD		
3 (1)	20	D16,D17,D18,D19,D20,D21	6 DD		2741 W
4 (1)	21	D22,D23,D24,D25,D26,D27	6 DD		5199 W
	22	---			
	23	---			

Modem # (#adrs)	Octal Modem Address	Traffic Device #	# Inputs	# Outputs	Approx ft W or E of Head-end
5 (4)	24	C3	8	8	7309 W
	25	C3	8 (stat)		
	26	D28,D29,D30,D36,D37	5 SD		
	27	---			
6 (4)	30	C4	8	8	8233 W
	31	C4	8 (stat)		
	32	D31,D32,D33,D34,D35	5 SD		
	33	D38,D39,D40,D41,D42,D43	6 DD		
7 (1)	34	D44,D45,D46,D47,D48,D49	6 DD		10618 W
8 (1)	35	D50,D51,D52,D53,D54,D55	6 DD		13020 W
9 (1)	36	D56,D57,D58,D59,D60,D61	6 DD		16039 W
10 (1)	37	D62,D63,D64,D65,D66,D67	6 DD		18277 W
11 (4)	40	C5	8	8	20667 W
	41	C5	8 (stat)		
	42	D68,D69,D70,D77	4 SD		
	43	D71,D72,D73,D74,D75,D76	6 DD		
*12 (1)	44	D78,D79,D80,D81,D82 D83,D840,D84	8 DD		23100 W
13 (1)	45	sign controller	RS232C	RS232C	24308 W
*14 (1)	46	D85,D86,D87,D88,D89,D90	6 DD		25846 W
	47	---			
*15 (4)	50	C6	8	8	29142 W
	51	C6	8 (stat)		
	52	D91,D92,D93,D94,D95, D96,D97,D98	8 SD		
	53	D99,D100,D101,D103 D103,D104	6 DD		
*16 (1)	54	D105,D106,D107,D108 D109,D110,D111,D112	8 DD		32390 W
	55	---			
	56	---			
	57	---			
*17 (4)	60	C7	8	8	34520 W
	61	C7	8 (stat)		
	62	D113,D114,D115,D116	4 SD		
	63	D117,D118,D119,D120 D121,D122	6 DD		

Modem # (#adrs)	Octal Modem Address	Traffic Device #	# Inputs	# Outputs	Approx ft W or E of Head-end
*18 (4)	64	D123,D124,D125,D126, D127,D128	6 DD		37717 W
	65	D129,D130,D131,D132, D133,D134	6 DD		
	66	D135,D136	2 SD		
	67	---			
*19 (1)	70	D137,D138,D139,D140, D141,D142	6 DD		40201 W
*20 (1)	71	D143,D144,D145,D146, D147,D148	6 DD		42760 W
	72	---			
	73	---			
	74	---			
	75	---			
	76	---			
	77	--- (reserved for system timing btw W & E)			
*21 (1)	100	D501,D502,D503,D504, D505,D506	6 DD		2285 E
*22 (1)	101	D507,D508,D509,D510, D511, D512	6 DD		5058 E
*23 (1)	102	D519,D520	2 SD		6390 E
	103	---			
*24 (4)	104	C8	8	8	7744 E
	105	C8	8 (stat)		
	106	D513,D514,D515,D516, D517,D518	6 SD		
	107	D521,D522,D523,D524, D525,D526	6 DD		
*25 (1)	110	D527,D528,D529,D530, D531,D532	6 DD		10651 E
reset addr.	220	(for reset on 224 octal [148 decimal])			

Appendix 13/ Cable Delay Modification

Prepared by Larry Deck and Checked by Tom Bennett

This document is on floppy "MTC" (1006) [30,24]CABLEDELA.DOC

Introduction

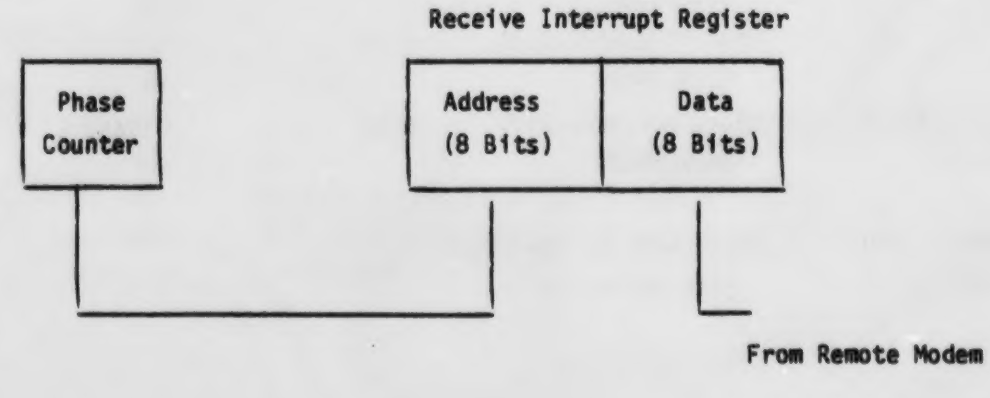
The cable delay modification is a design change, standard on new head-ends, which allows the head-end to control longer cables.

Cable delay is the time it takes for a signal, traveling at approximately the speed of light, to go from the head-end to a remote modem and back. Maximum cable lengths in miles, with and without the modification, are approximately as follows:

Baud Rate ---	48 000	24 000	12 000	(Etc.)
Cable length without mod	6	12	24	
Cable length with mod	20	40	80	

The modification is standard on new head-ends with transmit card Model number 23038 (BM78-8029-5077-8) and receive card Model number 23150 (BM78-8029-5081-2).

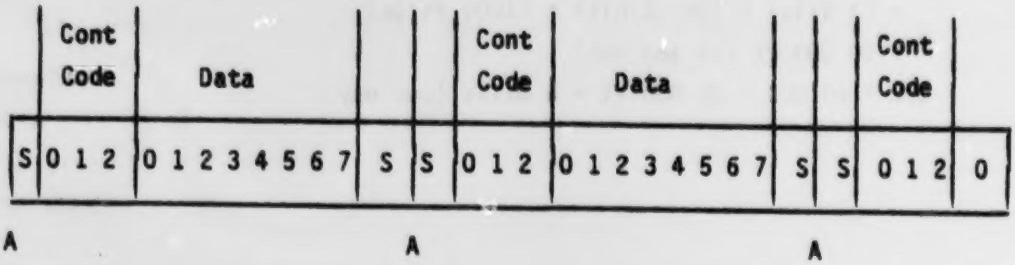
Relevant Headend Components



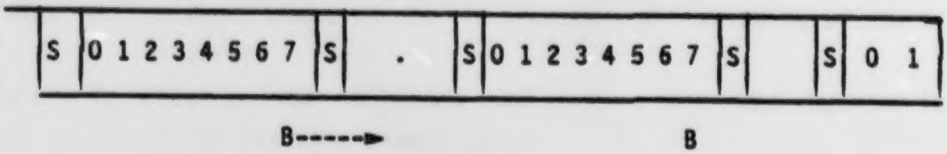
The head-end's phase counter is appended to the input data from a remote modem to tell the computer which modem address the data came from. The cable delay modification changes the time at which the phase counter is loaded into the receive interrupt register.

Timing Without Modification

Transmit
Data
(13 Bits
Per Word)


Cont Code
Data
Cont Code
Data
Cont Code
S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0
A
A
A

Receive
Data
(10 Bits
Per Word)


S 0 1 2 3 4 5 6 7 S . S 0 1 2 3 4 5 6 7 S S 0 1
B----►
B

Point A: Head-end phase counter is incremented (beginning of start bit of TX word).

Point B: Receive interrupt register is loaded with input data and current value of phase counter (middle of last input bit), and received interrupt is generated.

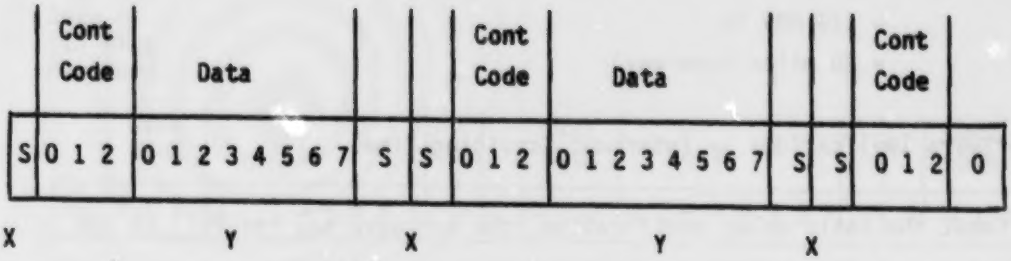
B--► A: Maximum cable delay -- 3 bit times.

If there is no cable delay, the input data is offset by 1/2 bit time from the transmit data, as indicated. As a cable becomes longer, the input data from a modem will return later with respect to the head-end's transmit data. That is, the receive data will slide to the right, and point "B" will slide closer to Point "A". If "B" slides past "A", the phase counter will have been incremented, and the input byte will have the wrong address appended. Thus the cable delay must be less than 3 bit times. Example:

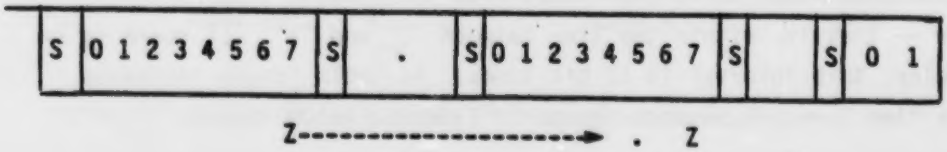
Signal speed on cable = 1000 ft/μs
Bit time at 48 000 baud = 20 μs/bit
Maximum cable length
  = (3 bits) * (20 μs/bit) * (1000 ft/μs)
  = 60 000 ft (in and out)
1/2 * 60 000 = 30 000 ft = 6 miles (one way)

Timing With Modification

Transmit
Data
(13 Bits
Per Word)


Cont Code
Data
Cont Code
Data
Cont Code
S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0
X
Y
X
Y
X

Receive
Data
(10 Bits
Per Word)


S 0 1 2 3 4 5 6 7 S . S 0 1 2 3 4 5 6 7 S S 0 1
Z--------------------→ . Z

Point X: Head-end phase counter is incremented (beginning of start bit TX word).

Point Y: Phase counter is loaded into receive interrupt register (end of fourth bit of transmit data).

Point Z: Input data is loaded into receive interrupt register (middle of last input bit), and receive interrupt is generated.

Z--→ Y: Maximum cable delay --- 11 bit times.

Input point "Z" may slide 11 bit times to point "Y" before the receive interrupt register is loaded with the next address. Example (48 000 baud):

Maximum cable length
= 1/2 * (11 bits) * (20 μs/bit) * (1000 ft/μs)
= 110 000 ft
= 20 miles (one way)

Software Implications -- Interrupt Servicing Time

Without the cable delay modification, the software has the full 13 bit times to service a receive interrupt (that is, until the next input byte is received). Example (48 000 baud):

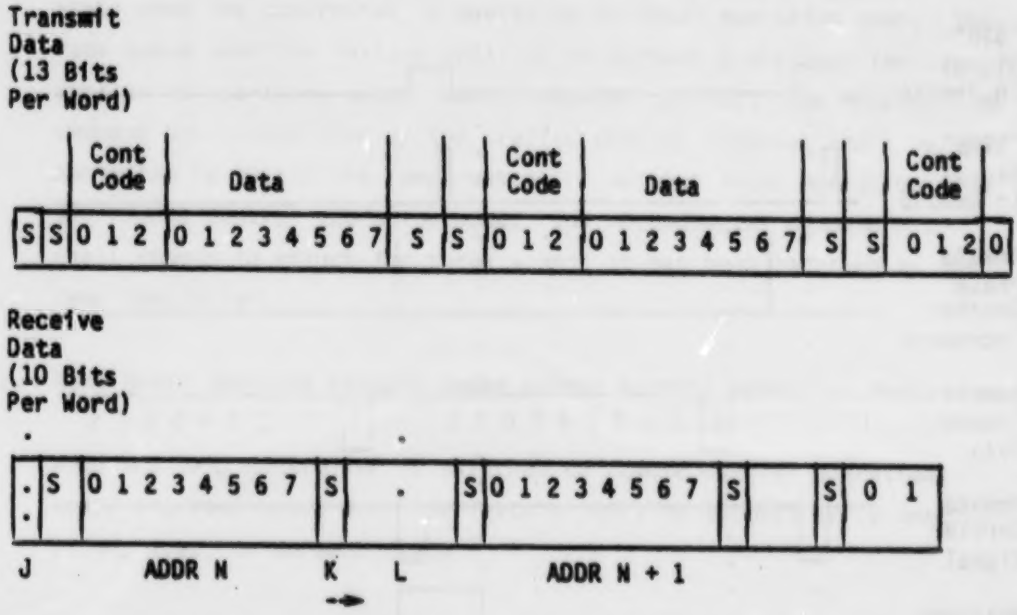
13 bits/48 000 bits per s = 270 μs

With the modification, the software must service a receive interrupt before the phase counter is pre-loaded into the receive interrupt register -- that is, within the time between "Z" and "Y". If there is no cable delay, this interval is 11 bit times. As cable length increases, response time Z→ Y becomes shorter. Example (48 000 baud):

11 bits/48 000 bits per s = 229 μs maximum

Distance Between Consecutive Address

If an address is substantially closer to the head-end than the address before it, their RF carrier signals may be on at the same time, causing problems.


Transmit Data (13 Bits Per Word)
Cont Code
Data
Cont Code
Data
Cont Code
S S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0
Receive Data (10 Bits Per Word)
S 0 1 2 3 4 5 6 7 S
S 0 1 2 3 4 5 6 7 S
S 0 1
J
ADDR N
K
L
ADDR N + 1

Point J: Remote modem address "N" turns on its carrier.

Point K: Remote address "N" turns off its carrier (end of stop bit).

Point L: Remote address "N + 1" turns on its carrier.

K--> L: Maximum delay between addresses "N" and "N + 1"; that is, address "N" must not be more than 1 bit time farther from the head-end than address "N + 1".

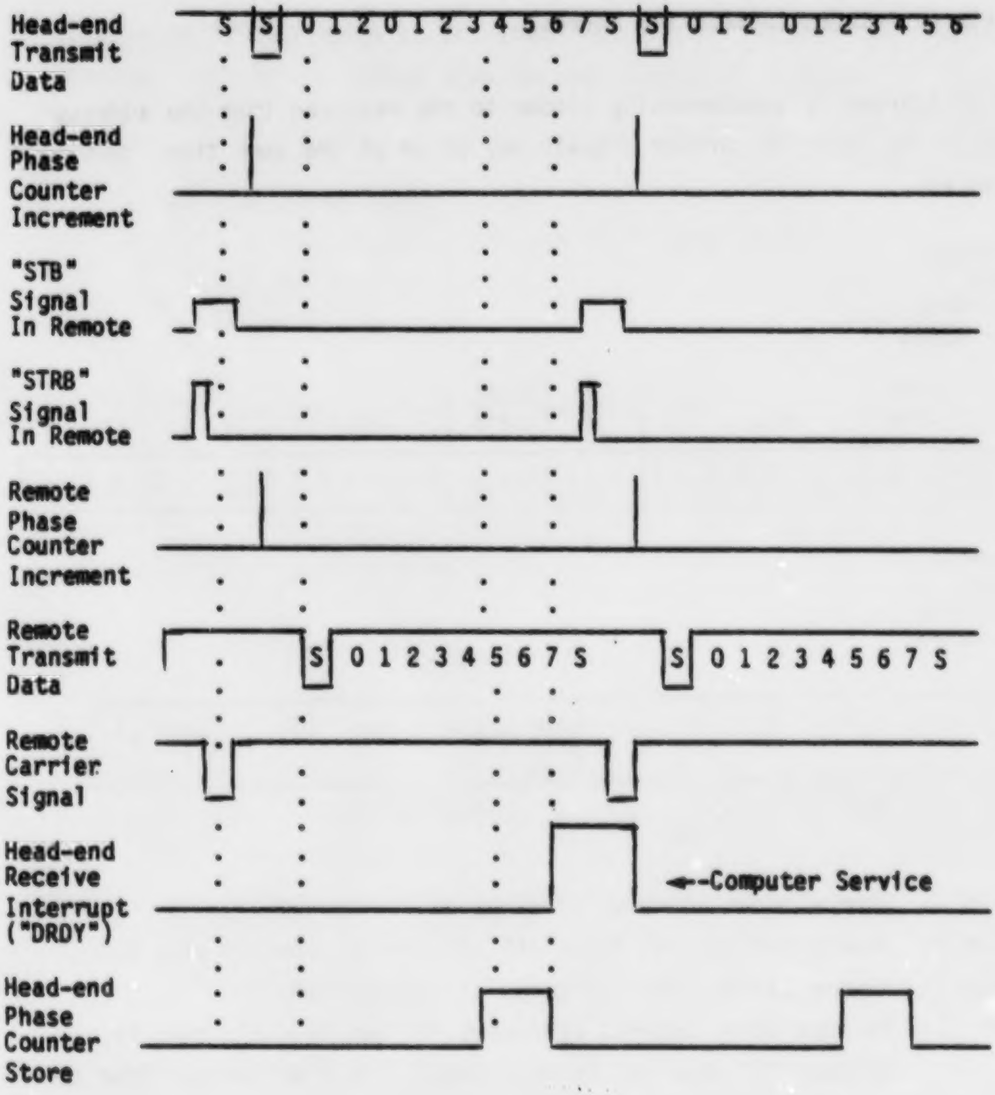
Example (48 000 Baud):

Maximum distance ADDR N may be beyond ADDR N + 1
= 1/2 * (1 bit) * (20 μs/bit) * (1000 ft/μs)
= 10 000 ft
= 2 miles (one way)

If addresses proceed sequentially from the head-end, this problem does not arise.

Note: The remote modem is able to turn on its carrier before receiving the start bit from the head-end. Because it has already incremented its own phase counter (see "overall system timing").

Overall System Timing (With Cable Delay Mod)


Head-end Transmit Data
S S 0 1 2 0 1 2 3 4 5 6 7 S S 0 1 2 0 1 2 3 4 5 6
Head-end Phase Counter Increment
"STB" Signal In Remote
"STRB" Signal In Remote
Remote Phase Counter Increment
Remote Transmit Data
S 0 1 2 3 4 5 6 7 S S 0 1 2 3 4 5 6 7 S
Remote Carrier Signal
Head-end Receive Interrupt ("DRDY")
←-Computer Service
Head-end Phase Counter Store

Notes:

1. The indicated events occur on bit or half-bit intervals (except for computer service).
2. "Computer Service" occurs when the program issues the instruction that clears the device flag.
3. When the remote phase counters increment, they are set to address "N + 1" before address "N" sends its data. Thus modem "N + 1" knows that it will be next, and can turn on its carrier before receiving the head-end's start bit.
4. Transmission delays due to a long cable cause the events in the remote modems to move to the right on the timing diagram.

Appendix 14/ CR710 Ramp Controller Ramp Queue Override Function

1. Switching Between Local Operation Mode and Remote Control Operation Mode

The ramp controller ramp queue override feature shall be in active state when the controller is operating in local operation mode. The ramp queue override feature shall be in dormant state when the controller is operating under remote computer control. The HOLD-ON-LINE command bit transmitted by the traffic control computer shall be the mechanism to switch the ramp controller between local operation mode and remote control operation mode. The HOLD-ON-LINE command bit shall remain in effect for three s once it has been received by the ramp controller.

2. Ramp Queue Override Feature Under Remote Control Operation Mode.

When the ramp controller is switched to remote control operation mode, the ramp queue override feature shall be automatically deactivated, i.e., in dormant state.

3. Ramp Queue Override Feature Under Local Control Operation Mode.

The ramp controller shall be able to input from one (1) to two (2) ramp queue detectors. If the occupancy of any one of the queue detectors is exceeding a preset threshold value T1, which is corresponding to the maximum allowable queue length, the the current cycle length of the ramp controller is changed to the next lowest cycle length. If the condition persists for "T3" seconds following the cycle length change, another cycle length reduction will take place. This cycle length reduction continues until the minimum cycle length is reached, or until the occupancy of the queue detector is decreasing to below a preset threshold value T2, which is corresponding to the minimum acceptable queue length. If the present controller cycle length is already operating at minimum cycle length, then the cycle length reduction shall not cause a non-metering condition.

After the queue detector occupancy is decreasing to below the threshold value T2, the metering rate of the ramp controller shall again respond to the mainline freeway occupancy level. If the queue detector fails, the queue override function shall be disabled.

4. Setting of Queue Override Function Threshold Value Tl, T2 and T3.

Due to ramp geometrics and traffic demand difference, it is necessary that the threshold occupancy values Tl, T2 and T3 can be set up in the ramp controller's variable function table through keyboard entries. When Tl and T2 are equal to zero, the ramp queue override feature shall be automatically in the dormant state regardless of whether the ramp controller is operating in the local control operation mode or remote control operating mode.

Appendix 15/ CR710 Interface Functional Assignment

Interface Requirements Between the QEW FSCS Traffic Control Computer and the CR710 Microprocessor Ramp Controller.

1. 8 - Status Bits to Traffic Control Computer.

Bit	Function
0	Occupancy status
1	Cycle length status
2	Cycle length status
3	Cycle length status
4	Demand pretimed
5	Passage pretimed
6	Spare
7	Spare

where

(1) Cycle length status
    000 - Non metering
    001 - Cycle length, 1A
    010 - Cycle length, 2A
    011 - Cycle length, 3A
    100 - Cycle length, 1B
    101 - Cycle length, 2B
    110 - Cycle length, 3B
    111 - Manual cycle length

(2) Demand pretimed
    0 - Pretimed
    1 - Actuated

(3) Passage pretimed
    0 - Pretimed
    1 - Actuated

(4) Occupancy
    0 - Occupancy not called for
    1 - Occupancy called for

2. 8 - Command Bits to Microprocessor Ramp Controller

There can be from one to four 8-bit command messages sent to microprocessor ramp controller.

Message 1

Bit	Function
0	Hold on Line (HOL)
1	Cycle length desired
2	Cycle length desired
3	Cycle length desired
4	Demand pretimed
5	Passage pretimed
6	"0"1st Message Address
7	"0"1st Message Address

Message 2

Bit	Function
0	HOL
1	Least Significant cycle length digit (BCD)
2	Least Significant cycle length digit (BCD)
3	Least Significant cycle length digit (BCD)
4	Least Significant cycle length digit (BCD)
5	- -
6	"1"2nd Message Address
7	"0"2nd Message Address

Message 3

Bit	Function
0	HOL
1	Middle cycle length digit (BCD)
2	Middle cycle length digit (BCD)
3	Middle cycle length digit (BCD)
4	Middle cycle length digit (BCD)
5	- -
6	"0"3rd Message Address
7	"1"3rd Message Address

Message 4

Bit	Function
0	HOL
1	Most significant cycle length digit (BCD)
2	Most significant cycle length digit (BCD)
3	Most significant cycle length digit (BCD)
4	Most significant cycle length digit (BCD)
5	- -
6	"1"4th Message Address
7	"1"4th Message Address

where

(1) HOL (The amount of time that the controller will remain on line without a command is set at 3 s.)
0 - Do Not Hold on Line
1 - Hold on Line

(2) Cycle length desired
Same code as cycle length status in Section 1.

(3) Demand pretimed
0 - Pretimed
1 - Actuated

(4) Passage pretimed
0 - Pretimed
1 - Actuated

(5) Message address
00 - 1st Message
01 - 2nd Message
10 - 3rd Message
11 - 4th Message

(6) Cycle length digits (BCD)

BCD	DECIMAL
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

Appendix 16/ Sample Sign Display and Sign Style

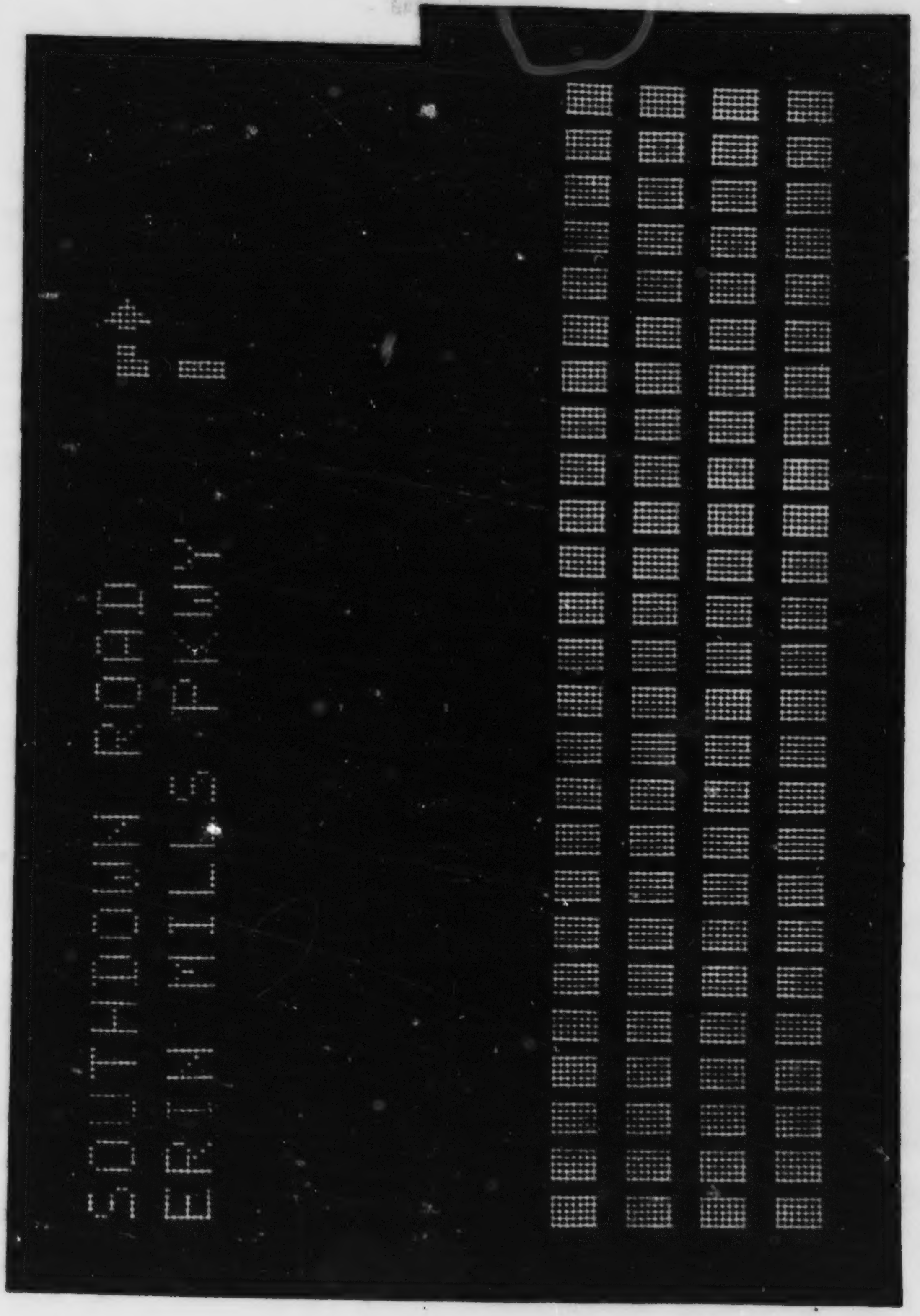

SOUTHDOWN ROAD
ERIN MILLS PKWY

Figure 7/ Sample Sign Display and Sign Style

Appendix 17/ CMS Sign Character Symbol and ASCII Code

Bits				b7 → b6 → b5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1
b4	b3	b2	b1	COLUMN → ROW ↓	0	1	2	3	4	5
0	0	0	0	0	NUL*	DLE*		0	■	P
0	0	0	1	1	SOH	DC1*	!	1	A	Q
0	0	1	0	2	STX	DC2*	"	2	B	R
0	0	1	1	3	ETX	DC3*	#	3	C	S
0	1	0	0	4	EOT	DC4*	$	4	D	T
0	1	0	1	5	ENQ*	NAK*	%	5	E	U
0	1	1	0	6	ACK*	SYN*	&	6	F	V
0	1	1	1	7	BEL	ETB*	)	7	G	W
1	0	0	0	8	BS	CAN*	◢	8	H	X
1	0	0	1	9	HT	EM*	◣	9	I	Y
1	0	1	0	10	LF	SUB*	↙	▪ ▪	J	Z
1	0	1	1	11	VT	ESC*	•••	▪	K	◀
1	1	0	0	12	FF*	FS*	◤	↓	L	◥
1	1	0	1	13	CR	GS*	—	⧗	M	▶
1	1	1	0	14	SO*	RS*	■	↘	N	—
1	1	1	1	15	SI*	US*	/	?	O	★

BEL - Cursor † home
BS - Cursor † left
HT - Cursor † right
SOH - Start of heading
STX - Start of text
ETX - End of text
EOT - End of transmission

LF - Cursor † down
VT - Cursor † up
CR - Cursor † to beginning of line

* - Character is ignored
† - Cursor indicates the character address position

Appendix 18/ CMS Controller Input Message Format and The CMS Controller Response.

The changeable message sign controller accepts the following input message format:

SOH $\underbrace{\text{BD}\quad\text{PNL}\quad\text{LN}}_{\text{Address portion}}$ MDL STX $\underbrace{\text{Data characters}}_{\text{Data portion}}$ ETX

where

SOH - start of heading. Upon receiving this character, the board, panel, line, and module addresses shall be reset to zero. The character address position moves to the upper left corner of the "BOARD".

BD - Board Address with the following ASCII address characters assigned:

P - Board 0
Q - Board 1
R - Board 2
S - Board 3
T - Board 4

PNL - Panel address with the following ASCII address characters assigned:

0 - Panel 0
1 - Panel 1
2 - Panel 2
3 - Panel 3
4 - Panel 4
5 - Panel 5
6 - Panel 6
7 - Panel 7

LN - Line address with the following ASCII address characters assigned:

@ - Line 0
A - Line 1
B - Line 2
C - Line 3
D - Line 4
E - Line 5
F - Line 6
G - Line 7
H - Line 8

I - Line 9
J - Line 10
K - Line 11
L - Line 12
M - Line 13
N - Line 14
O - Line 15

MDL - Module Address with the following ASCII address characters assigned:

0 - Module 0
1 - Module 1
2 - Module 2
3 - Module 3
4 - Module 4
5 - Module 5
6 - Module 6
7 - Module 7
8 - Module 8
9 - Module 9

STX - Start of text

ETX - Terminate the message. All following characters will be ignored until either an SOH or BEL is received.

The following special control characters shall be accepted by the controller when used in the Data Portion of the message.

BEL - This character shall be regarded as "Home" character by the sign controller. The Board address remains the same but PANEL, LINE, and MODULE addresses are reset to zero. The character address position moves to the upper left corner of the "BOARD".

CR - The "CARRIAGE RETURN" character shall set both PANEL and MODULE addresses to zero. The character address position shall move to the character at the extreme left of a line.

LF - The "LINE FEED" character shall increment the line address. If the "character address portion" is on the bottom line, it shall recycle to TOP LINE (LINE 0).

VT - This "VERTICAL TAB" character shall decrement the LINE address. If the "character address portion" is on the TOP LINE, it shall recycle to BOTTOM LINE.

BS - This "BACK SPACE" character shall decrement the MODULE address. The "character address position" shall move left one character module position. If the MODULE ADDRESS is 0, it shall decrement PANEL address and shall set MODULE address to 9. If the "character address position" is on PANEL 0, MODULE 0, the "character address position" shall recycle to the last PANEL, last MODULE on the LINE above the original line (right most character module, up one line). If the "character address position" is on PANEL 0, LINE 0, MODULE 0, the "character address position" shall recycle to last PANEL, last MODULE, and last LINE (lower right most character module in the board).

HT - This "HORIZONTAL TAB" character shall increment the MODULE address, (all functions shall be the opposite of BS item).

ENQ - This "ENQ" character shall serve as "blankout" control character. Upon receiving this character, the sign controller shall blank out the Changeable Message Sign.

Note: An automatic CR/LF shall be performed by the sign controller at the end of the last character on a LINE.

The present CMS control program utilizes the following message format:

SOH P 0 @ 0 STX .... CR LF .... ETX

or

SOH BEL .... CR LF .... ETX

Response of Sign Controller to "ENQ" and "SOH"

When the sign controller receives an "ENQ",

- the echo data line goes immediately to a "BREAK" condition -- the "ENQ" is not echoed.
- the "BREAK" clears automatically in 6 s, and the controller is then ready to receive more characters.

When the sign controller asserts a "BREAK" condition on the echo data line

- if the "BREAK" was due to a parity error, the sign is not blanked.
- if the "BREAK" was due to something other than a parity error, the sign is blanked.

When the sign controller receives an "SOH" while in a "BREAK" condition:

1. If the "BREAK" was due to a parity error:
    - the "BREAK" will clear immediately.
    - the "SOH" will be echoed back.

- the controller is ready for more characters immediately.

2. If the "BREAK" was due to something other than a parity error:
   - the "SOH" is not echoed
   - the "BREAK" persists after the "SOH" is received.

Computer-Computer Interface for "ENQ" and "SOH" Messages

"ENQ" and "SOH" messages from the host computer will be sent in the protocol 4 buffer only as 1 - byte message of the following form:

High BYTE	Low BYTE
0	ADDR
CHAR	0

where:

- "ADDR" is the modem address.
- "CHAR" is an "ENQ" or "SOH".

This message may be preceded by or followed by other messages in the protocol 4 buffer.

The response in the Protocol 5 buffer will be a 1-byte message of the same form, where "CHAR" is as defined below.

"ENQ" Message From Host

Echo From Sign Controller	"CHAR" In Prot 5 Buffer	Message Status
Break	NAK	0 (Correct)
CHAR other than Break	ENQ	1 (Error)
No response	0	1 (Error)

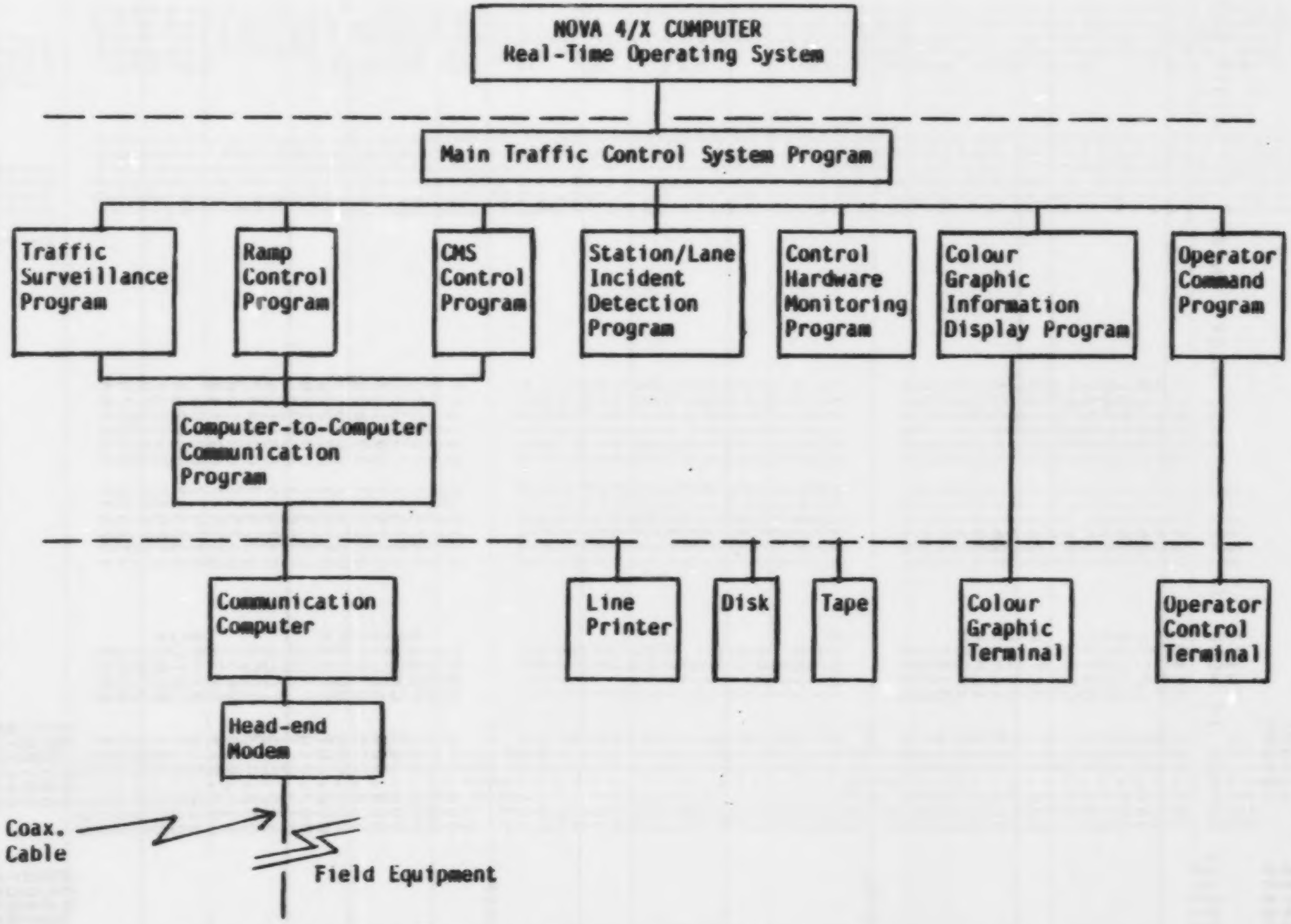
"SOH" Message From Host

The cable network processor assumes that the host will only send an "SOH" to try to clear a "BREAK", and processes the echoed message accordingly. When an "SOH" is sent, it will take about 5 s before the protocol 5 buffer is ready.

Echo From Sign Controller	"CHAR" In Prot 5 Buffer	Message Status
Break Clears	SOH	0 (Correct)
Break Persists	NAK	1 (Error)
Some other CHAR	That CHAR	1 (Error)

Appendix 19/ CMS Controller and Data Modem Interface

Pin Number	Description
2	Feedback or echo signal from sign controller to remote data modem
3	Character data signal from remote modem to sign controller
4	Request to send
7	Signal ground
8	Received Line Carrier signal Detector.


NOVA 4/X COMPUTER
Real-Time Operating System
Main Traffic Control System Program
Traffic Surveillance Program
Ramp Control Program
CMS Control Program
Station/Lane Incident Detection Program
Control Hardware Monitoring Program
Colour Graphic Information Display Program
Operator Command Program
Computer-to-Computer Communication Program
Communication Computer
Line Printer
Disk
Tape
Colour Graphic Terminal
Operator Control Terminal
Head-end Modem
Coax. Cable
Field Equipment

Appendix 20/ QEW FSCS REAL-TIME TRAFFIC CONTROL SOFTWARE FUNCTIONAL BLOCK DIAGRAM

Appendix 21/ QEW Real-Time Application Software Overlay Structure Chart Program Size

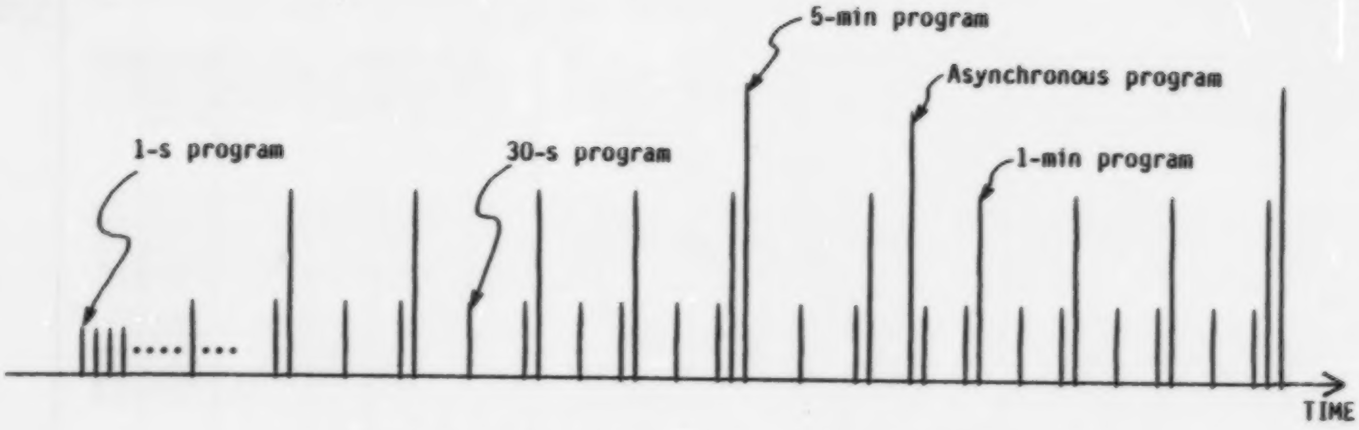
```
QEW.OV   LOADED BY RLDR REV 07.10 AT 11:50:03 05/03/83
 .MAIN      000566
 .MAIN      000675
 COMCO      012761
 DATAI      014304
 RAMCO      017035
 MODM1      021557
 SLCHE      022777
 OLCHE      023232
 NXFER      024057
 TRANS      025205
         026111
         000.000   INITI      026111
LABELED COMMON IN NODE DEFINED OUTSIDE NODE   NREC      00337? 000006
001072
         000.001   HEADI      026111 000671
         000.002   W6J03      026111 000476
         000.003   UPD3U      026111 002374
         000.004   INCDE      026111 001451
         000.005   PERCH      026111 001560
         000.006   DATST      026111 001021
         000.007   CLDU       026111 002367
         000.010   VTLU       026111 002624
         000.011   RCTIM      026111 002033
         000.012   BUGCC      026111 002207
         000.013   ALM30      026111 002323
         000.014   ALMSM      026111 003367
         000.015   AMODM      026111 001265
         000.016   DIS1       026111 001442
         000.017   DYS1       026111 001422
         000.020   DIS2       026111 002116
         031511
  TOPCO     031511
         033275
         001.000   INPED      033275 001433
         001.001   T12A       033275 001301
         001.002   T12B       033275 000742
         001.003   T13        033275 001621
         001.004   T14A       033275 001375
         001.005   T14B       033275 001775
         001.006   T14C       033275 001325
         001.007   T14D       033275 001560
         001.010   T15        033275 000473
         001.011   T16A       033275 001637
         001.012   T16B       033275 001765
         001.013   T16C       033275 001705
         001.014   T16D       033275 001647
         001.015   T17A       033275 001772
         001.016   T17B       033275 001374
         001.017   T18A       033275 001661
         001.020   T18B       033275 001301
         001.021   T18C       033275 001674
         035275
  GMAP      035275
         037537
         002.000   QSPD       037537 001077
         002.001   QOCC       037537 000444
         002.002   REQEW      037537 000070
         002.003   CHQOR      037537 000327
         002.004   ACCID      037537 000163
         002.005   DISMA      037537 000626
         002.006   RDISM      037537 000623
         002.007   FLASH      037537 001246
         002.010   FACC       037537 000126
         002.011   LOOP       037537 000326
         002.012   IOCC       037537 001166
         002.013   IOC2       037537 000432
         002.014   SPD1       037537 001162
         002.015   SPD2       037537 001143
         002.016   RAMPS      037537 001067
         002.017   DETEC      037537 000264
         002.020   PCBOX      037537 000507
         002.021   BOX1       037537 000274
         002.022   BOX2       037537 000575
         002.023   BOX3       037537 000464
         041137
  ALMSG     041137
  PCSPO     041260
  MISCO     041267
  CMSCT     041276
  SORST     041305
  ARTSI     041314
  CLOCK     041323
  ISDEF     041340
  MTI       041471
```

Appendix 21/ (Cont'd)

```
ITASK     042360
FPRI      042405
FXMT      042413
MOVLD     042444
MOVRL     U42462
STTSK     U42474
DESTR     042503
FCNS      042512
QUCLK     042526
DUCLK     042534
FSPOL     042544
WAIT      042574
TIME      042656
UATE      042670
FREAD     042705
THREA     047064
HOFLD     047130
READL     047240
DFTAB     047513
READR     047616
FFILE     047672
UPEN      047751
OFILW     050234
CFILW     050257
CLOSE     050362
IOPRE     050374
ITEST     050412
BSTRI     050463
FREDI     050540
FALOC     050644
ARYSZ     050664
FSBR      050702
GCT       051077
MOD       051123
SMPY      051134
SDVD      051160
MNMXD     051226
IPWER     051272
AFRTN     051345
MVBT      051352
LDD       051411
STOP      051430
ISARE     051454
FLINK     051467
RTER      051633
WRCH      052046
BDASC     052063
BASC      052144
GOUT      052200
LDSTB     052223
CPYAR     052253
MAD       052315
FARGO     052340
ARGUM     052373
FPZER     052443
DUMMY     052443
ARDUM     052443
HMPYD     052444
TOVLY     052455
IXMT      052721
TXMT      052721
TIDC      053056
TPRI      053216
TABT      053241
MTUMO     053336
TSVRS     053366
SMTSK     053514
KILAD     053527
MTCBM     053527
NSAC3     054201
DUMMY     054201

      NMAX      054210
      ZMAX      000217
      CSZE      000000
       EST      000000
       SST      000000
```

AUTOREPORT


1-s program
30-s program
5-min program
Asynchronous program
1-min program
TIME

Appendix 22/ Real-Time Multi-Task Program Scheduling Sequence

Appendix 23/ "Detect" Traffic Data Buffer

In the traffic data buffer, raw input data from each modem is followed by volume (VOL) and elapsed time (ET) calculations for each detector point (DX) attached to that modem. For double detectors, the double detector elapsed time (DDET) follows the VOL and ET calculations for each detector pair. For modems attached to double detectors, the high byte of each word storing input data is marked with * to indicate that the byte is used by the BDCSP program for storing flags used in processing double detector elapsed time.

Each input point is attached to one bit in the input byte from a modem. VOL and ET calculations proceed in order from least significant bit to most significant bit. For example, for modem address 12, D1 is assigned to bit 0, D2 to bit 1, D5 to bit 2, and D7 to bit 3.

```
                                             Modem
             High     Low                    address   Input        # Input
Word #       byte     byte                   (octal)   type         points
------       ----     ----                   -------   -----        -------
         15        8  7         0
        <---------------+---------------->
   1    ! Load     C1 ! [illegible]   C1 !   10-11     Controller   (8)
   2    ! switch   C2 ! [illegible]   C2 !   14-15     inputs
   3    ! status   C3 !               C3 !   24-25
   4    !          C4 !               C4 !   30-31
   5    !          C5 !               C5 !   40-41
   6    !          C6 !               C6 !   50-51
   7    !          C7 !               C7 !   60-61
   8    !          C8 !               C8 !   100-101
        <---------------+---------------->
   9    !             ! Input data      !    12        Single       (4)
        <---------------+---------------->             detector
  10    !                VOL - D1       !              inputs
  11    !                ET  - D1       !
  12    !                VOL - D2       !
  13    !                ET  - D2       !
  14    !                VOL - D5       !
  15    !                ET  - D5       !
  16    !                VOL - D7       !
  17    !                ET  - D7       !
        <---------------+---------------->
  18    !             ! Input data      !    16        Single       (5)
        <---------------+---------------->             detector
  19    !                VOL - D3       !
  20    !                ET  - D3       !
  21    !                VOL - D4       !
  22    !                ET  - D4       !
  23    !                VOL - D6       !
  24    !                ET  - D6       !
  25    !                VOL - D8       !
  26    !                ET  - D8       !
  27    !                VOL - D9       !
  28    !                ET  - D9       !
```

Appendix 23/ (Cont'd)

```
     <-------------+-------------->
 29  !             ! Input data !      26     Single             (5)
     <-------------+-------------->               detector
 30  !                VOL - D28  !
 31  !                ET  - D28  !
 32  !                VOL - D29  !
 33  !                ET  - D29  !
 34  !                VOL - D30  !
 35  !                ET  - D30  !
 36  !                VOL - D36  !
 37  !                ET  - D36  !
 38  !                VOL - D37  !
 39  !                ET  - D37  !
     <-------------+-------------->
 40  !             ! Input data !      32     Single             (5)
     <-------------+-------------->               detector
 41  !                VOL - D31  !
 42  !                ET  - D31  !
 43  !                VOL - D32  !
 44  !                ET  - D32  !
 45  !                VOL - D33  !
 46  !                ET  - D33  !
 47  !                VOL - D34  !
 48  !                ET  - D34  !
 49  !                VOL - D35  !
 50  !                ET  - D35  !
     <-------------+-------------->
 51  !             ! Input data !      42     Single             (4)
     <-------------+-------------->               detector
 52  !                VOL - D68  !
 53  !                ET  - D68  !
 54  !                VOL - D69  !
 55  !                ET  - D69  !
 56  !                VOL - D70  !
 57  !                ET  - D76  !
 58  !                VOL - D77  !
 59  !                ET  - D77  !
```

Appendix 23/ (Cont'd)

```
       <-------------------+-------------------->
 60    !                   ! Input data !        52      Single            (8)
       <-------------------+-------------------->        detector
 61    !                      VOL - D91 !
 62    !                      ET  - [illegible] !
 63    !                      VOL - D92 !
 64    !                      ET  - D92 !
 65    !                      VOL - D93 !
 66    !                      ET  - D93 !
 67    !                      VOL - D94 !
 68    !                      ET  - [illegible] !
 69    !                      VOL - [illegible] !
 70    !                      ET  - [illegible] !
 71    !                      VOL - D96 !
 72    !                      ET  - D96 !
 73    !                      VOL - D97 !
 74    !                      [illegible] !
 75    !                      [illegible] !
 76    !                      [illegible] !
       <-------------------+-------------------->
 77    !                   ! Input data !        62      Single            (4)
       <-------------------+-------------------->        detector
 78    !                      VOL - [illegible]
 79    !                      ET  - [illegible]
 80    !                      VOL - [illegible]
 81    !                      ET  - D114!
 82    !                      VOL - D115!
 83    !                      ET  - D115!
 84    !                      [illegible]
 85    !                      [illegible]
       <-------------------+-------------------->
 86    !                   ! Input data !        66      Single            (2)
       <-------------------+-------------------->        detector
 87    !                      VOL - D135!
 88    !                      ET  - [illegible]
 89    !                      VOL - [illegible]
 90    !                      ET  - [illegible]
       <-------------------+-------------------->
 91    !                   ! Input data !        102     Single            (2)
       <-------------------+-------------------->        detector
 92    !                      VOL - D319!
 93    !                      ET  - D319!
 94    !                      VOL - D320!
 95    !                      ET  - D320!
```

```
        <-------------+------------->
  96    !             ! Input data !     106      Single              (6)
        <-------------+------------->             detector
  97    !               VOL  - D513!
  98    !               ET   - D513!
  99    !               VOL  - D514!
 100    !               ET   - D514!
 101    !               VOL  - D515!
 102    !               ET   - D515!
 103    !               VOL  - D516!
 104    !               ET   - D516!
 105    !               VOL  - D517!
 106    !               ET   - D517!
 107    !               VOL  - D518!
 108    !               ET   - D518!
        <-------------+------------->
 109    !      *      ! Input data !     17       Double              (3 pairs)
        <-------------+------------->             detector
 110    !               VOL  - D10 !
 111    !               ET   - D10 !
 112    !               VOL  - D11 !
 113    !               ET   - D11 !
 114    !          DDET - D10, D11 !
 115    !               VOL  - D12 !
 116    !               ET   - D12 !
 117    !               VOL  - D13 !
 118    !               ET   - D13 !
 119    !          DDET - D12, D13 !
 120    !               VOL  - D14 !
 121    !               ET   - D14 !
 122    !               VOL  - D15 !
 123    !               ET   - D15 !
 124    !          DDET - D14, D15 !
        <-------------+------------->
 125    !      *      ! Input data !     [illegible]  Double          (3 pairs)
        <-------------+------------->             detector
 126    !               VOL  - D16 !
 127    !               ET   - D16 !
 128    !               VOL  - D17 !
 129    !               ET   - D17 !
 130    !          DDET - D16, D17 !
 131    !               VOL  - D18 !
 132    !               ET   - D18 !
 133    !               VOL  - D19 !
 134    !               ET   - D19 !
 135    !          DDET - D18, D19 !
 136    !               VOL  - D20 !
 137    !               ET   - D20 !
 138    !               VOL  - D21 !
 139    !               ET   - D21 !
 140    !          DDET - D20, D21 !
```

```
      <----------------+---------------->
141   !       *        ! Input data     !      21      Double          (3 pairs)
      <----------------+---------------->              detector
142   !                      VOL - D22  !
143   !                      ET  - D22  !
144   !                      VOL - D23  !
145   !                      ET  - D23  !
146   !                DDET - D22, D23  !
147   !                      VOL - D24  !
148   !                      ET  - D24  !
149   !                      VOL - D25  !
150   !                      ET  - D25  !
151   !                DDET - D24, D25  !
152   !                      VOL - D26  !
153   !                      ET  - D26  !
154   !                      VOL - D27  !
155   !                      ET  - D27  !
156   !                DDET - D26, D27  !
      <----------------+---------------->
157   !       *        ! Input data     !      33      Double          (3 pairs)
      <----------------+---------------->              detector
158   !                      VOL - D38  !
159   !                      ET  - D38  !
160   !                      VOL - D39  !
161   !                      ET  - D39  !
162   !                DDET - D38, D39  !
163   !                      VOL - D40  !
164   !                      ET  - D40  !
165   !                      VOL - D41  !
166   !                      ET  - D41  !
167   !                DDET - D40, D41  !
168   !                      VOL - D42  !
169   !                      ET  - D42  !
170   !                      VOL - D43  !
171   !                      ET  - D43  !
172   !                DDET - D42, D43  !
      <----------------+---------------->
173   !       *        ! Input data     !      34      Double          (3 pairs)
      <----------------+---------------->              detector
174   !                      VOL - D44  !
175   !                      ET  - D44  !
176   !                      VOL - D45  !
177   !                      ET  - D45  !
178   !                DDET - D44, D45  !
179   !                      VOL - D46  !
180   !                      ET  - D46  !
181   !                      VOL - D47  !
182   !                      ET  - D47  !
183   !                DDET - D46, D47  !
184   !                      VOL - D48  !
185   !                      ET  - D48  !
186   !                      VOL - D49  !
187   !                      ET  - D49  !
188   !                DDET - D48, D49  !
```

```
             +-----------------+-------------+
 189         !   *             ! Input data  !    35     Double          (3 pairs)
             +-----------------+-------------+           detector
 190         !                   VOL  - D50  !
 191         !                   ET   - D50  !
 192         !                   VOL  - D51  !
 193         !                   ET   - D51  !
 194         !            DDET - D50, D51    !
 195         !                   VOL  - D52  !
 196         !                   ET . - D52  !
 197         !                   VOL  - D53  !
 198         !                   ET   - D53  !
 199         !            DDET - D52, D53    !
 200         !                   VOL  - D54  !
 201         !                   ET   - D54  !
 202         !                   VOL  - D55  !
 203         !                   ET   - D55  !
 204         !            DDET - D54, D55    !
             +-----------------+-------------+
 205         !   *             ! Input data  !    36     Double          (3 pairs)
             +-----------------+-------------+           detector
 206         !                   VOL  - D56  !
 207         !                   ET   - D56  !
 208         !                   VOL  - D57  !
 209         !                   ET   - D57  !
 210         !            DDET - D56, D57    !
 211         !                   VOL  - D58  !
 212         !                   ET   - D58  !
 213         !                   VOL  - D59  !
 214         !                   ET   - D59  !
 215         !            DDET - D58, D59    !
 216         !                   VOL  - D60  !
 217         !                   ET   - D60  !
 218         !                   VOL  - D61  !
 219         !                   ET   - D61  !
 220         !            DDET - D60, D61    !
             +-----------------+-------------+
 221         !   *             ! Input data  !    37     Double          (3 pairs)
             +-----------------+-------------+           detector
 222         !                   VOL  - D62  !
 223         !                   ET   - D62  !
 224         !                   VOL  - D63  !
 225         !                   ET   - D63  !
 226         !            DDET - D62, D63    !
 227         !                   VOL  - D64  !
 228         !                   ET   - D64  !
 229         !                   VOL  - D65  !
 230         !                   ET   - D65  !
 231         !            DDET - D64, D65    !
 232         !                   VOL  - D66  !
 233         !                   ET   - D66  !
 234         !                   VOL  - D67  !
 235         !                   ET   - D67  !
 236         !            DDET - D66, D67    !
```

237	*	Input data	43	Double detector	(3 pairs)
238		VOL - D71			
239		ET - D71			
240		VOL - D72			
241		ET - D72			
242		DDET - D71,D72			
243		VOL - D73			
244		ET - D73			
245		VOL - D74			
246		ET - D74			
247		DDET - D73,D74			
248		VOL - D75			
249		ET - D75			
250		VOL - D76			
251		ET - D76			
252		DDET - D75,D76			
253	*	Input data	48	Double detector	(4 pairs)
254		VOL - D78			
255		ET - D78			
256		VOL - D79			
257		ET - D79			
258		DDET - D78,D79			
259		VOL - D80			
260		ET - D80			
261		VOL - D81			
262		ET - D81			
263		DDET - D80,D81			
264		VOL - D82			
265		ET - D82			
266		VOL - D83			
267		ET - D83			
268		DDET - D82,D83			
269		VOL - D84C	(dummy)		
270		ET - D84D	(dummy)		
271		VOL - D84			
272		ET - D84			
273		DDET - [illegible],D84	(dummy)		

```
             <-------------+--------------->
274          !      *      ! Input data    !      46      Double          (3 pairs)
             <-------------+--------------->              detector
275          !                  VOL - D85  !
276          !                  ET  - D85  !
277          !                  VOL - D86  !
278          !                  ET  - D86  !
279          !           DOET - D85,D86    !
280          !                  VOL - D87  !
281          !                  ET  - D87  !
282          !                  VOL - D88  !
283          !                  ET  - D88  !
284          !           DOET - D87,D88    !
285          !                  VOL - D89  !
286          !                  ET  - D89  !
287          !                  VOL - D90  !
288          !                  ET  - D90  !
289          !           DOET - D89,D90    !
             <-------------+--------------->
290          !      *      ! Input data    !      53      Double          (3 pairs)
             <-------------+--------------->              detector
291          !                  VOL - D99  !
292          !                  ET  - D99  !
293          !                  VOL - D100 !
294          !                  ET  - D100 !
295          !         DOET -  D99,D100    !
296          !                  VOL - D101 !
297          !                  ET  - D101 !
298          !                  VOL - D102 !
299          !                  ET  - D102 !
300          !         DOET - D101,D102    !
301          !                  VOL - D103 !
302          !                  ET  - D103 !
303          !                  VOL - D104 !
304          !                  ET  - D104 !
305          !         DOET - D103,D104    !
```

Appendix 23/ (Cont'd)

```
         <------------------+------------------>
306      !        *         ! Input data !          54      Double                  (4 pairs)
         <------------------+------------------>            detector
307      !                        VOL - D105!
308      !                        ET  - D105!
309      !                        VOL - D106!
310      !                        ET  - D106!
311      !                DDET - D105,D106!
312      !                        VOL - D107!
313      !                        ET  - D107!
314      !                        VOL - D108!
315      !                        ET  - D108!
316      !                DDET - D107,D108!
317      !                        VOL - D109!
318      !                        ET  - D109!
319      !                        VOL - D110!
320      !                        ET  - D110!
321      !                DDET - D109,D110!
322      !                        VOL - D111!
323      !                        ET  - D111!
324      !                        VOL - D112!
325      !                        ET  - D112!
326      !                DDET - D111,D112!
         <------------------+------------------>
327      !        *         ! Input data !          [illegible]  Double             (3 pairs)
         <------------------+------------------>            detector
328      !                        VOL - D117!
329      !                        ET  - D117!
330      !                        VOL - D118!
331      !                        ET  - D118!
332      !                DDET - D117,D118!
333      !                        VOL - D119!
334      !                        ET  - D119!
335      !                        VOL - D120!
336      !                        ET  - D120!
337      !                DDET - D119,D120!
338      !                        VOL - D121!
339      !                        ET  - D121!
340      !                        VOL - D122!
341      !                        ET  - D122!
342      !                DDET - D121,D122!
```

```
     +-----------------+-----------------+
343  !       *         ! Input data      !    64     Double            (3 pairs)
     +-----------------+-----------------+           detector
344  !                      VOL  - D123!
345  !                      ET   - [illegible]
346  !                      VOL  - [illegible]
347  !                      ET   - [illegible]
348  !              DDET - D123, D124!
349  !                      VOL  - D125!
350  !                      ET   - D125!
351  !                      VOL  - [illegible]
352  !                      ET   - [illegible]
353  !              DDET - [illegible]
354  !                      VOL  - D127!
355  !                      ET   - D127!
356  !                      VOL  - D128!
357  !                      ET   - [illegible]
358  !              DDET - [illegible]
     +-----------------+-----------------+
359  !       *         ! Input data      !    65     Double            (3 pairs)
     +-----------------+-----------------+           detector
360  !                      VOL  - D129!
361  !                      ET   - [illegible]
362  !                      VOL  - [illegible]
363  !                      ET   - [illegible]
364  !              DDET - D129, D130!
365  !                      VOL  - D131!
366  !                      ET   - D131!
367  !                      VOL  - [illegible]
368  !                      ET   - [illegible]
369  !              DDET - [illegible]
370  !                      VOL  - D133!
371  !                      ET   - D133!
372  !                      VOL  - D134!
373  !                      ET   - [illegible]
374  !              DDET - [illegible]
     +-----------------+-----------------+
375  !       *         ! Input data      !    70     Double            (3 pairs)
     +-----------------+-----------------+           detector
376  !                      VOL  - D137!
377  !                      ET   - [illegible]
378  !                      VOL  - D138!
379  !                      ET   - D138!
380  !              DDET - D137, D138!
381  !                      VOL  - D139!
382  !                      ET   - D139!
383  !                      VOL  - D140!
384  !                      ET   - D140!
385  !              DDET - D139, D140!
386  !                      VOL  - D141!
387  !                      ET   - D141!
388  !                      VOL  - D142!
389  !                      ET   - D142!
390  !              DDET - D141, D142!
```

Appendix 23/ (Cont'd)

```
          +---------------+-------------+
 391      !      *        ! Input data  !      71      Double              (3 pairs)
          +---------------+-------------+              detecter
 392      !                 VOL  - D143!
 393      !                 ET   - [illegible]
 394      !                 VOL  - D[illegible]!
 395      !                 ET   - D146!
 396      !          DDET - D143.D144!
 397      !                 VOL  - D145!
 398      !                 ET   - D145!
 399      !                 VOL  - [illegible]
 400      !                 ET   - [illegible]
 401      !          DDET - [illegible]
 402      !                 VOL  - D147!
 403      !                 ET   - D147!
 404      !                 VOL  - D148!
 405      !                 ET   - [illegible]
 406      !          DDET - [illegible]
          +---------------+-------------+
 407      !      *        ! Input data  !     100      Double              (3 pairs)
          +---------------+-------------+              detecter
 408      !                 VOL  - D501!
 409      !                 ET   - [illegible]
 410      !                 VOL  - [illegible]
 411      !                 [illegible]
 412      !          DDET - D501.D502!
 413      !                 VOL  - D503!
 414      !                 ET   - D503!
 415      !                 [illegible]
 416      !                 [illegible]
 417      !          DDET - [illegible]
 418      !                 VOL  - D505!
 419      !                 ET   - D505!
 420      !                 VOL  - D506!
 421      !                 ET   - [illegible]
 422      !          DDET - [illegible]
          +---------------+-------------+
 423      !      *        ! Input data  !     101      Double              (3 pairs)
          +---------------+-------------+              detecter
 424      !                 VOL  - D507!
 425      !                 ET   - D507!
 426      !                 VOL  - D[illegible]!
 427      !                 ET   - D[illegible]!
 428      !          DDET - D507.D508!
 429      !                 VOL  - D509!
 430      !                 ET   - D509!
 431      !                 VOL  - D510!
 432      !                 ET   - D510!
 433      !          DDET - [illegible].D510!
 434      !                 VOL  - D511!
 435      !                 ET   - D511!
 436      !                 VOL  - D512!
 437      !                 ET   - D512!
 438      +          DDET - D511.D512!
```

Appendix 23/ (Cont'd)

```
         <-------------+-------------->
439      !     *       ! Input data   !      107      Double              (3 pairs)
         <-------------+-------------->               detector
440      !                VOL  - D521!
441      !                ET   - D521!
442      !                VOL  - D522!
443      !                ET   - D522!
444      !         DDET - D521,D522!
445      !                VOL  - D523!
446      !                ET   - D523!
447      !                VOL  - D524!
448      !                ET   - D524!
449      !         DDET - D523,D524!
450      !                VOL  - D525!
451      !                ET   - D525!
452      !                VOL  - D526!
453      !                ET   - D526!
454      !         DDET - D525,D526!
         <-------------+-------------->
455      !     *       ! Input data   !      110      Double              (3 pairs)
         <-------------+-------------->               detector
456      !                VOL  - D527!
457      !                ET   - D527!
458      !                VOL  - D528!
459      !                ET   - D528!
460      !         DDET - D527.D528!
461      !                VOL  - D529!
462      !                ET  .- D529!
463      !                VOL  - D530!
464      !                ET   - D530!
465      !         DDET - D529,D530!
466      !                VOL  - D531!
467      !                ET   - D531!
468      !                VOL  - D532!
469      !                ET   - D532!
470      !         DDET - D531,D532!
```

Appendix 23/ (Cont'd)

471	Modem response counts	0
472		1
473		2
474		3
475		4
476		5
477		6
478		7
479		10
480		11
481		12
482		13
483		14
484		15
485		16
486		17
487		20
488		21
489		22
490		23
491		24
492		25
493		26
494		27
495		30
496		31
497		32
498		33
499		34
500		35
501		36
502		37
503		40
504		41
505		42
506		43
507		44
508		45
509		46
510		47
511		50
512		51
513		52
514		53
515		54
516		55
517		56
518		57
519		60
520		61
521		62
522		63
523		64
524		65

```
325    !                                  !          66
326    !                                  !          67
327    !                                  !          70
328    !                                  !          71
329    !                                  !          72
330    !                                  !          73
331    !                                  !          74
332    !                                  !          75
333    !                                  !          76
334    !                                  !          77
335    !                                  !          100
336    !                                  !          101
337    !                                  !          102
338    !                                  !          103
339    !                                  !          104
340    !                                  !          105
341    !                                  !          106
342    !                                  !          107
343    !                                  !          110
       <---------------------------------->
344    ! Sign echo message status         !
       <---------------------------------->
345    !        Transfer count            !
       <---------------------------------->
```

Appendix 24/ "COMDOP" Ramp Control Command Buffer Format

Word	Left Byte	Right Byte	On-Ramp
1	Control data	Modem Address	Cawthra NB & SB
2	Control data	Modem Address	Highway 10 NB
3	Control data	Modem Address	Highway 10 SB
4	Control data	Modem Address	Mississauga NB
5	Control data	Modem Address	Mississauga SB
6	Control data	Modem Address	Southdown Road
7	Control data	Modem Address	Winston NB & SB
8	Control data	Modem Address	Ford Drive

The eight-words buffer, sent from the traffic control computer, to the communication computer, hold control information for the ramp controllers. The left byte of each word holds the control data, and the right byte holds the modem address to receive that data. The communication computer precedes every word sent to a modem with a control code 3, which resets its local time out counter. The traffic control computer sends output to a modem at least every 3 s to prevent its controller from going into local control.

Appendix 25/ Computerized Ramp Metering Rate Selection Logic

1/ Introduction

Based on the experience from the implemented Minneapolis freeway control system and the recommendation from the project consultant Dr. Brian L. Allen, the following coordinated ramp metering rate selection logic is drafted. This logic is intended to be implemented as the first phase ramp control logic on the traffic control computer (Nova 800 system). It is expected that this computerized ramp control logic will be improved as the experience is gained from the real system operation.

2/ Traffic Control Section

Due to the fact that each interchange is unique in geometrics, demand, and relation to the entire traffic corridor, the freeway is divided into three (3) control sections according to the interchange boundary. The purpose of the control section is to determine the location of the decision variable, i.e., volume and occupancy, for each entrance ramp control strategy. Figure 1 shows the proposed traffic control section (adapted from consultant proposal) according to the availability of the hardware facilities. From Figure 1, it is clear that the metering rate of an entrance ramp will not only be determined by the nearby local detector station data but will also be determined by its upstream and downstream detector station data. Basically, this is to achieve the function of the coordinated ramp control system.

3/ Ramp Characteristics Table

For the purpose of the ramp metering rate selection within a control section, a characteristics table (RAMTAB) for all ramps which are available for entrance control will be made up of the station numbers of the ramp's local, upstream and downstream detector stations. These station numbers will be used as the index pointer to access the real-time detector data and ramp cycle length selection table.

Figure 2 shows the system ramp characteristics table. The numbers shown in the Figure 2 is an example of the traffic control section configuration.

4/ Ramp Cycle Length Selection Table

The numerical threshold values used for real-time selection of metering rates are stored in the table 'CYCTAB' as shown in Figure 3. These values are the thresholds of occupancy for the local and downstream detector stations, the volume thresholds of upstream detector stations for each of the three control sections, and the associated cycle lengths for each of six different metering rates.

The values shown in the Figure 3 is an example of the mainline detector station volume and occupancy threshold value. Since the 'occupancy' is only an indication of the freeway congestion status, it is felt that the same set of the local station occupancy or downstream station occupancy threshold values can be used for all five entrance ramps. If one set of the occupancy threshold values is required for each entrance ramp control, the table 'CYCTAB' can be expanded to cover this situation.

5/ Real-time Mainline Detector Station Data Table

The mainline detector station data used for metering rate selection will be the latest 1-min running average occupancy by station and the latest 1-min running average volume by station. These 1-min running average volume and occupancy data will be updated every 30 s. Figure 4 shows the latest 1-min running average occupancy and volume table. Note that the data in the table will be changing depending on the real-time traffic situation.

6/ Ramp Controller Metering Rate Setting

The CR710 ramp controller's six-cycle length metering rates, which are selectable from the traffic control computer, will be pre-set locally according to each ramp's characteristics. For example, the setting of the CR710 controller's six metering rates can be made according to whether the ramp is a low volume, average volume, or high volume entrance ramp. In other words, it can be made according to the cumulative ramp demand function for each entrance ramp. Figure 5 shows the example of the metering rate setting for the five ramp controllers.

7/ Ramp Control Period

It is generally agreed and practiced that the ramp control period be divided into three periods. In the period 1, a ramp will be turned on only if it is warranted by the mainline traffic flow, that is, the real-time data exceeding the lowest volume or occupancy threshold value. In the period 2, all the entrance ramps shall be available for coordinated ramp metering control. In the period 3, a ramp will be turned off as mainline traffic flow drops off. Once a ramp is turned on, the rate at which vehicles are allowed to enter the freeway will remain responsive to the mainline traffic unless a traffic controller metering rate override command is exercised. In the latter case, the metering rate override command will remain effective until it is cancelled. This is a feature which can allow for the relief of the excessive ramp queue delay.

8/ The Selection of the Metering Rate

Throughout the control period when the ramp is active, the metering rate selection process will take place every 30 s, and the metering rate is changed if warranted by the mainline real-time traffic conditions.

As described in the preceding paragraph, there are three decision variables to be used for metering rates selection process, namely, local station occupancy, downstream station occupancy, and upstream station volume.

The actual selection process can be best illustrated by an example. For instance, if the metering rate selection process is to be performed on the Highway 10 Southbound entrance ramp, the associated local station number '9' is obtained from ramp characteristics table

'RAMTAB'. This local station number '9' is used as an index to obtain the latest 1-min running average occupancy data from mainline detector station data table (Figure 4). This real-time occupancy value, for example, 27%, is then compared with the occupancy threshold value in the ramp cycle length selection table 'CYCTAB' (Figure 3). It is found that '27%' falls between the threshold value '23%' and '30%'. Therefore, the corresponding cycle length '4' will be selected.

The same selection process will be repeated for the associated downstream station number '10'. For example, if the real-time occupancy value at the downstream station number '10' is found to be '35%', then the cycle length '5' will be chosen from the table 'CYCTAB'.

For the associated upstream station number '7', if the real-time volume value is found to be '78' vehicles, then the corresponding cycle length '3' will be selected from the table 'CYCTAB' because the Highway 10 Southbound ramp is in control section 3 (C.S.3).

It is obvious now that using three decision variables, i.e., local station occupancy, downstream station occupancy, and upstream station volume can result in three different cycle lengths to be chosen. In this example, they are cycle length '3', '4', and '5'.

In the case of isolated ramp control system (essentially only local station occupancy is used as decision variable), the cycle length '4' will be chosen.

The Minneapolis freeway control system will choose the longest cycle length, that is, '5'. This is a conservative approach to prevent the congestion from happening in the mainline freeway.

The project consultant, Dr. Allen, proposes to assign each decision process a weighting factor, then choosing the maximum one, or simply taking the average value; in this example, it will be cycle length '4'. Apparently, if the main objective of the freeway ramp control is to prevent the freeway congestion which could be due to excessive upstream volume influx, excessive local occupancy, or excessive downstream occupancy, then choosing the longest cycle length seems to be a logical approach.

In any event, if the selected cycle length is '5' for example, it will be transmitted by the communication system to the Highway 10 southbound ramp controller. The controller will then choose the 9-s as the next cycle length from its pre-set metering rate value (see Figure 5).

9/ Synchronization of The Ramp Controller and Traffic Control Computer

To avoid the problem that the CR710 ramp controller is dropped off-line too long, it is necessary to adjust the local timer in the CR710 controller to set-up the same ramp control period as used by the traffic control computer.

10/ Handling of the Entrance Ramp Queue

Since the excessive ramp queue delay is undesirable, provision will be made to relieve the excessive ramp queue length by metering rate override command from the computer console teletypewriter.

However, the local ramp controller shall also possess the ramp queue override feature which shall be automatically activated when it drops off-line from the computer system. This feature shall also be automatically de-activated when the controller goes on-line (connected to the computer system) such that only the computer can exercise the ramp queue override feature.

11/ Flexibility of the Ramp Metering Rate Selection Logic

Since this is a demonstration project, flexibility will be provided to allow for changing of the traffic control section, ramp characteristics table and cycle length selection table, such that different control concepts can be exercised and tested.

It is also expected that as a more sophisticated control algorithm is developed, the preceding ramp metering rate selection logic will be replaced.

TIME - HH:MM:SS/DD:MM:YY
WEATHER -

(Appendix 25)

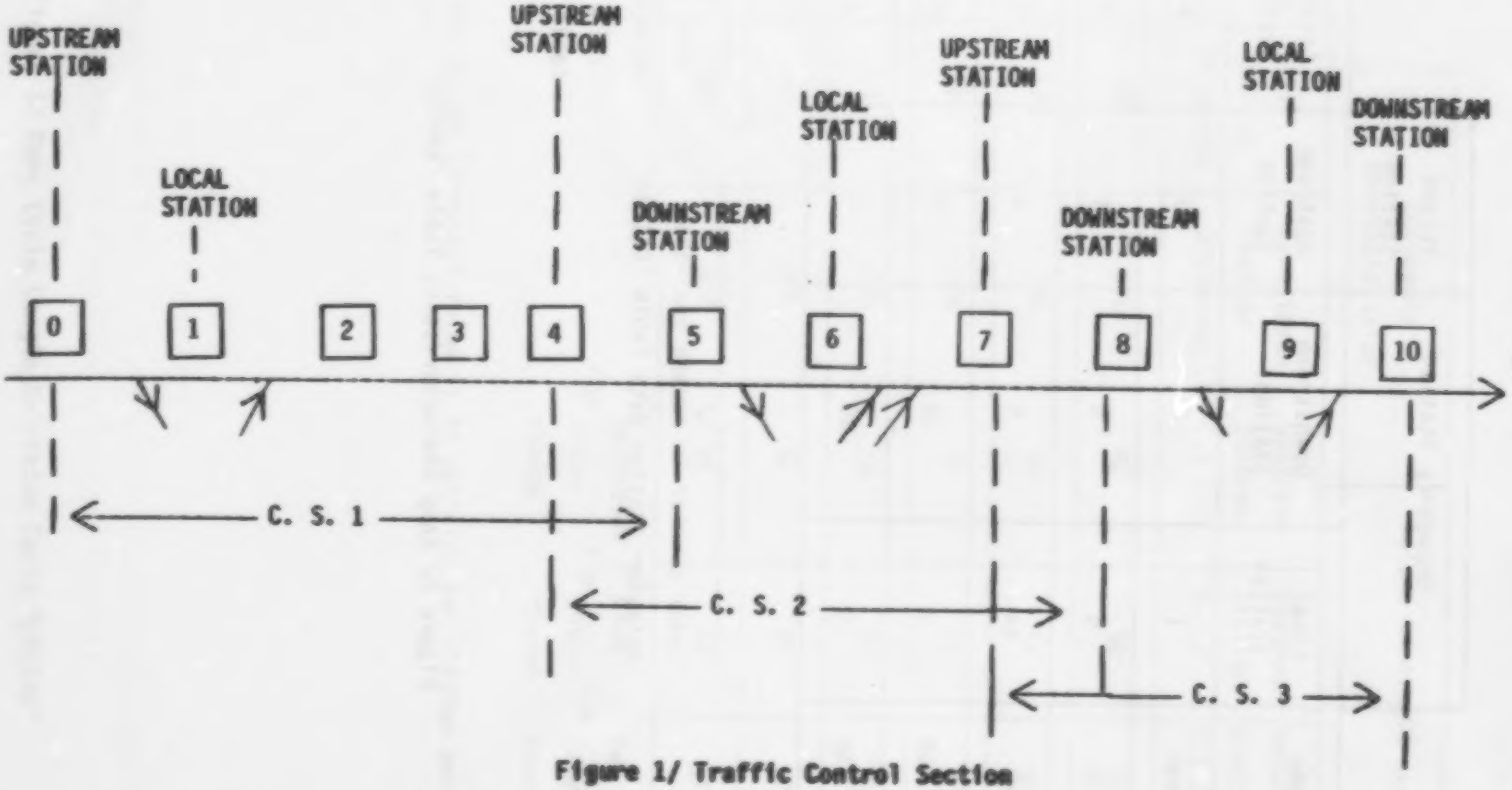

UPSTREAM STATION
LOCAL STATION
UPSTREAM STATION
DOWNSTREAM STATION
LOCAL STATION
UPSTREAM STATION
DOWNSTREAM STATION
LOCAL STATION
DOWNSTREAM STATION
0
1
2
3
4
5
6
7
8
9
10
QEW
C. S. 1
C. S. 2
C. S. 3

Figure 1/ Traffic Control Section

(Appendix 25)

	Occupancy Station		Volume Station	
Ramp Name	Local Station	Downstream Station	Upstream Station	Control Section 3
Southdown	1	5	0	4
Miss. SB	6	8	4	5
Miss. NB	6A	8	4	5
Hwy 10. SB	9	10	7	6
Hwy 10. NB	9	10	7	6

Index pointer to access real-time detector station data Table 'OCCTAB'

Index pointer to access ramp cycle length table 'CYCTAB'

Figure 2/ Ramp Characteristics Table "RAMTAB"

(Appendix 25)

	Occupancy threshold value		Volume threshold value		
Selected Cycle Length	Local Station Occupancy	Downstream Station Occupancy	Upstream Station C.S.1	Upstream Station C.S.2	Upstream Station C.S.3
1	10%	10%	47	58	60
2	13%	13%	55	67	65
3	16%	16%	63	72	73
4	23%	23%	73	77	81
5	30%	30%	82	80	87
6	60%	60%	85	84	91

Southdown Interchange

Miss. Interchange

HWY 10 Interchange

Note the the volume threshold value is in vehicles per station per minute.

Figure 3/ Ramp Cycle Length Selection Table "CYCTAB"

(Appendix 25)

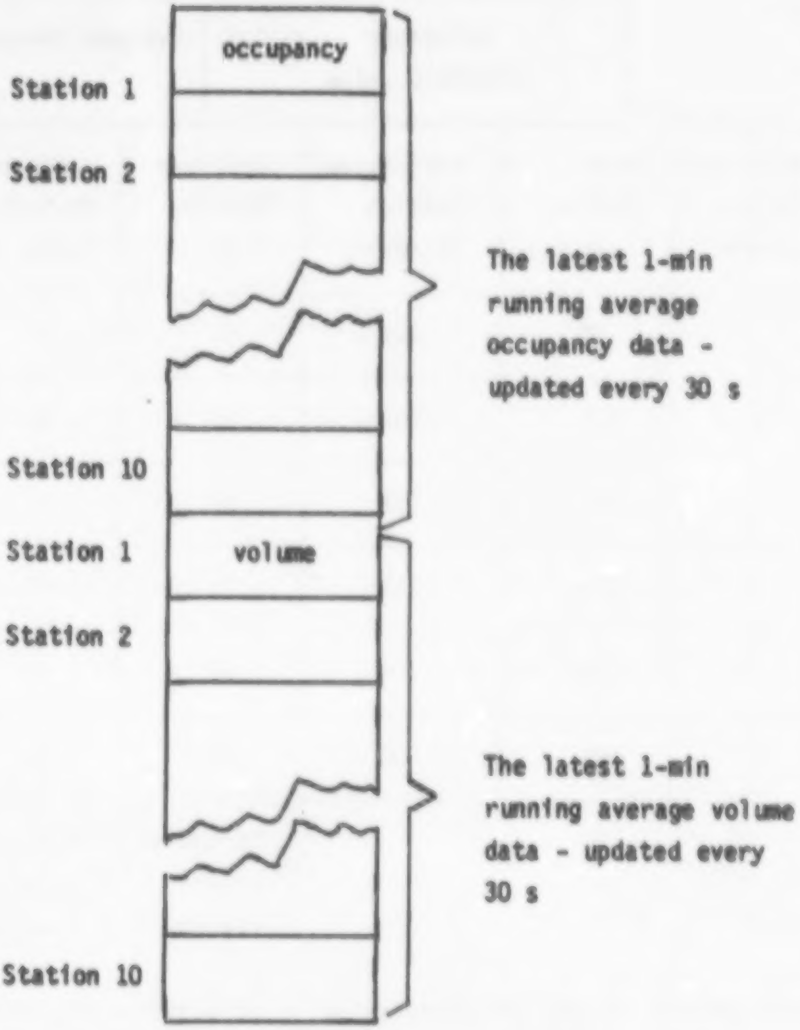

occupancy
Station 1
Station 2
The latest 1-min running average occupancy data - updated every 30 s
Station 10
Station 1
volume
Station 2
The latest 1-min running average volume data - updated every 30 s
Station 10

Figure 4/ Mainline Detector Station Data Table

(Appendix 25)

	Cycle Length					
Ramp Name	1	2	3	4	5	6
Southdown	5.0	6.0	8.0	12.0	12.9	15.0
Miss. SB	5.5	6.9	9.0	12.9	13.9	14.4
Miss. NB	5.0	6.2	8.0	11.6	12.4	13.3
HWY 10 SB	4.5	5.7	7.2	8.0	9.0	11.6
HWY 10 NB	5.5	6.9	9.0	13.3	14.4	15.0

- an example -

Note that the value is in seconds

Figure 5/ The Metering Rate to be pre-set in the CR710 Ramp Controller (An example. Note that the value is in seconds)

Appendix 26/ Computerized Ramp Queue Detection Logic and Ramp Queue Override Feature

In a computerized freeway ramp control system, in order not to penalize the ramp users waiting excessively on the ramp, the ramp queue length detection and ramp queue override feature are a necessity.

1/ Ramp Queue Detector Location

Figures 1, 2 and 3 show the ramp queue detector location for three controlled QEW interchanges. Table 1 shows the estimated distance between each ramp queue detector and its corresponding demand detector.

2/ Ramp Queue Length Detection Logic

Obviously, to detect ramp queue length between the queue detector and the demand detector for a certain interval, e.g., every 30 s, it is only necessary to subtract the 30-s demand detector volume from the 30-s queue detector volume. The preceding procedure will generate the correct ramp queue length provided that the initial ramp queue length is equal to zero.

The detection of initial ramp queue length equal to zero can be accomplished by checking the start-up queue gap time for each queue detector before a ramp is turned on for metering.

Note that before a ramp is turned on for metering, the signal light is permanently green. Therefore, the vehicle does not have to stop on demand detector waiting for green signal. In other words, the traffic on the ramp is in free flow condition before morning rush hour metering period. Under this condition, the time interval required for a vehicle to travel from queue detector to the corresponding demand detector can be estimated or measured fairly accurately. This time interval will be called start-up queue gap time - Tg.

The logic to detect the initial ramp queue length equal to zero is then simply monitoring the queue detector in "OFF" state for at least the period of time which is equal to the corresponding start-up queue gap time - Tg. Table 1 shows each ramp's estimated start-up queue gap time based on the assumption that the average ramp speed is 30 km/h.

When the start-up queue gap time is detected, the computer will initialize the corresponding ramp queue length according to the succeeding volume count of the queue detector until the beginning of the next 30-second time slice period (assuming the traffic data is accumulated every 30 s.)

From then on, the ramp queue length is simply equal to the sum of the preceding 30-s time slice ramp queue length plus the difference between the current 30-s queue detector volume and the current 30-s demand detector volume.

3/ Ramp Queue Override Feature

Table 1 also shows the estimated maximum allowable ramp queue length for each entrance ramp. When the ramp is turned on for metering, it is possible that the ramp queue length will approach the maximum allowable queue length due to excessive ramp demand. When this situation occurs, the traffic controller can override the computer selected metering rate by traffic control operator command language. It is estimated that with a 5 s cycle length, the maximum queue length can be cleared up within 6 min. Automatic ramp queue override feature can be incorporated in the future should it become necessary.

4/ Occupancy as Ramp Queue Length Detection Threshold

Since the 'occupancy' is an indication of the roadway congestion status, it is possible to use the queue detector occupancy as a threshold value to determine whether the maximum allowable ramp queue length has been reached or not. However, this can only be achieved after real field data has been collected and analysed.

Table 1/ Queue Detector Characteristics Table

Appendix 26 (Continued)

Location	Q-detector	Distance	Maximum Queue Length	Start-up Queue Gap Time
HWY 10 (NB)	Q1	600'	24 veh.	20 s
HWY 10 (SB)	Q2	1 700'	68 veh.	55 s
MISS. (NB)	Q3	550'	22 veh.	18 s
MISS. (SB)	Q4	800'	32 veh.	26 s
MISS. (SB)	Q5	830'	33 veh.	28 s
SOUTHD. (NB)	Q6	1 250'	50 veh.	41 s
SOUTHD. (SB)	Q7	1 180'	47 veh.	39 s

(Appendix 26)

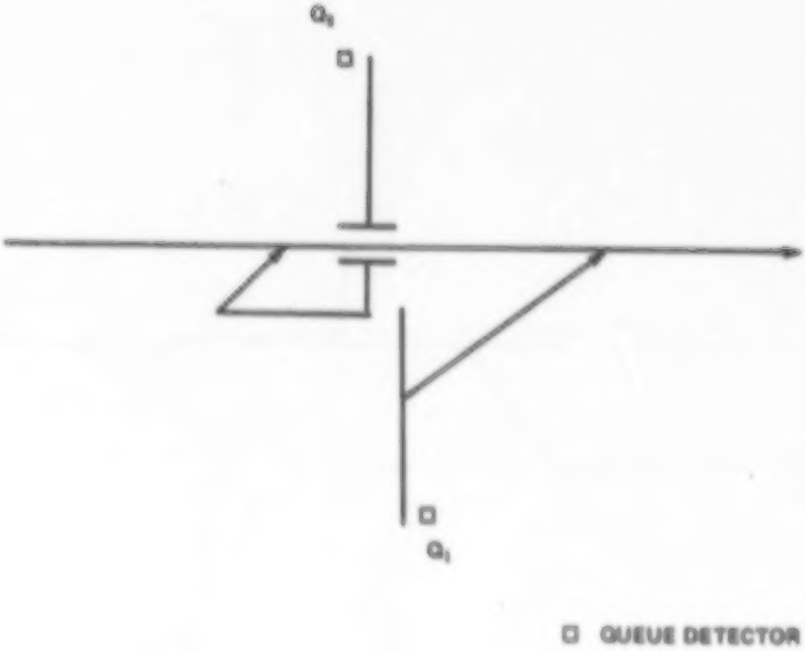
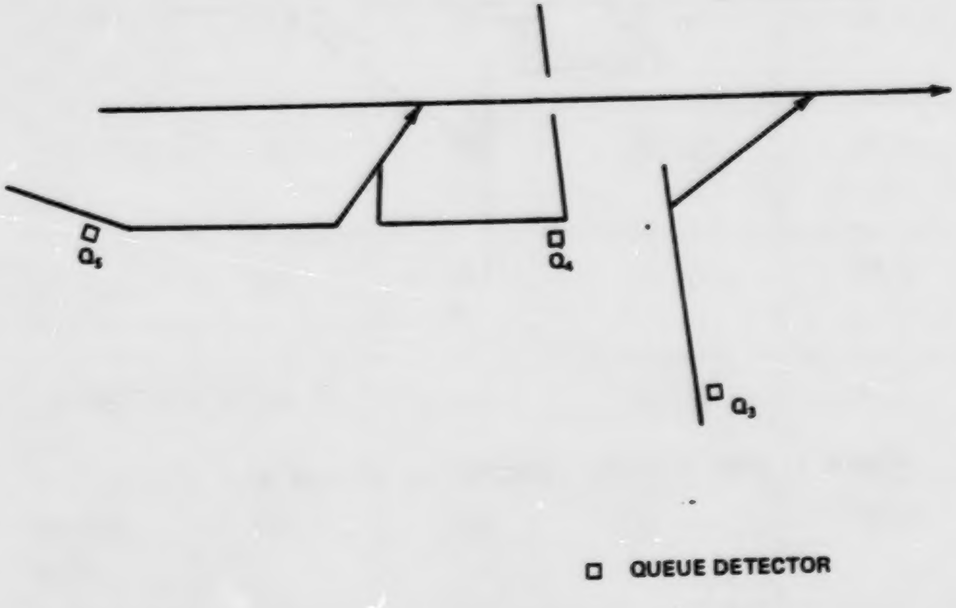

Q_2
Q_1
□ QUEUE DETECTOR

Figure 1/ Ramp Detector Locations at Highway 10

(Appendix 26)



Q_5
Q_4
Q_3
□ QUEUE DETECTOR

Figure 2/ Ramp Detector Locations at Mississauga Road

(Appendix 26)

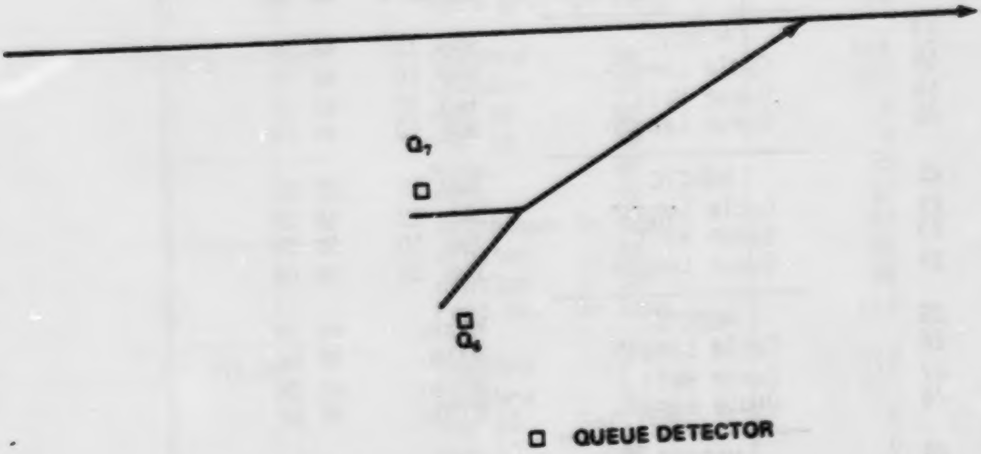

Q_7
Q_6
□ QUEUE DETECTOR

Figure 3/ Ramp Detector Locations at Southown Road

Appendix 27/ Data30 File Logical Record

Word#	Content	Station#	Lane#	Time Slice	Detector #
1	Hour				
2	min			30-s	
3	s			interval	
4	Control Period				
5	Reserved			Reserved	
6	Reserved			for HOV	
7	Reserved				
8	Reserved				
9	MODCYC	Cawthra	NB ON	30-s	
10	Cycle Length	Cawthra	NB ON		
11	Queue Wait	Cawthra	NB ON		
12	Queue Length	Cawthra	NB ON		
13	MODCYC	Cawthra	SB ON		
14	Cycle Length	Cawthra	SB ON		
15	Queue Wait	Cawthra	SB ON		
16	Queue Length	Cawthra	SB ON		
17	MODCYC	Hwy 10	NB ON		
18	Cycle Length	Hwy 10	NB ON		
19	Queue Wait	Hwy 10	NB ON		
20	Queue Length	Hwy 10	NB ON		
21	MODCYC	Hwy 10	SB ON		
22	Cycle Length	Hwy 10	SB ON		
23	Queue Wait	Hwy 10	SB ON		
24	Queue Length	Hwy 10	SB ON		
25	MODCYC	Miss.	NB ON		
26	Cycle Length	Miss.	NB ON		
27	Queue Wait	Miss.	NB ON		
28	Queue Length	Miss.	NB ON		
29	MODCYC	Miss.	SB ON		
30	Cycle Length	Miss.	SB ON		
31	Queue Wait	Miss.	SB ON		
32	Queue Length	Miss.	SB ON		
33	MODCYC	Southdown	ON		
34	Cycle Length	Southdown	ON		
35	Queue Wait	Southdown	ON		
36	Queue Length	Southdown	ON		
37	MODCYC	Winston	NB ON		
38	Cycle Length	Winston	NB ON		
39	Queue Wait	Winston	NB ON		
40	Queue Length	Winston	NB ON	30-s	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
41	MODCYC	Winston	SB ON	30-s	
42	Cycle Length	Winston	SB ON		
43	Queue Wait	Winston	SB ON		
44	Queue Length	Winston	SB ON		
45	MODCYC	Ford Dr.	ON		
46	Cycle Length	Ford Dr.	ON		
47	Queue Wait	Ford Dr.	ON		
48	Queue Length	Ford Dr.	ON		
49	PASSAGE VOLUME	Cawthra	NB		513
50		Cawthra	SB		516
51		Hwy 10	NB		1
52		Hwy 10	SB		3
53		Miss.	NB		28
54		Miss.	SB		31
55		Southdown	On Ramp		68
56		Winston	NB		91
57		Winston	SB		94
58		Ford Dr.	On Ramp		113
59	DEMAND VOLUME	Cawthra	NB		514
60		Cawthra	SB		517
61		Hwy 10	NB		2
62		Hwy 10	SB		4
63		Miss.	NB		29
64		Miss.	SB		32
65		Southdown	On Ramp		69
66		Winston	NB		92
67		Winston	SB		95
68		Ford Dr.	On Ramp		114
69	EXIT VOLUME	Cawthra	1		519
70		Cawthra	2		520
71		Hwy 10	Service		5
72		Hwy 10	NB		6
73		Hwy 10	SB		9
74		Miss.	Off Ramp		35
75		Southdown	Off Ramp		84
76		Winston	1		97
77		Winston	2		98
78		Ford Dr.	1		135
79		Ford Dr.	2		136
80	QUEUE VOLUME	Cawthra	NB		515
81		Cawthra	SB		518
82		Hwy 10	NB		7
83		Hwy 10	SB		8
84		Miss.	NB		30
85		Miss.	SBE		33
86		Miss.	SBW	30-s	34

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
87	QUEUE VOLUME	Southdown	S	30-s	70
88		Southdown	N		77
89		Winston	NB		93
90		Winston	SB		96
91		Ford Dr.	S		115
92		Ford Dr.	N		116
93	VOL	15A	Driving		36
94	VOL	15A	Centre		37
95	PASSAGE OCCUPANCY	Cawthra	NB		513
96		Cawthra	SB		516
97		Hwy 10	NB		1
98		Hwy 10	SB		3
99		Miss.	NB		28
100		Miss.	SB		31
101		Southdown	On Ramp		68
102		Winston	NB		91
103		Winston	SB		94
104		Ford Dr.	On Ramp		113
105	DEMAND OCCUPANCY	Cawthra	NB		514
106		Cawthra	SB		517
107		Hwy 10	NB		2
108		Hwy 10	SB		4
109		Miss.	NB		29
110		Miss.	SB		32
111		Southdown	On Ramp		69
112		Winston	NB		92
113		Winston	SB		95
114		Ford Dr.	On Ramp		114
115	EXIT OCCUPANCY	Cawthra	1		519
116		Cawthra	2		520
117		Hwy 10	Service		5
118		Hwy 10	NB		6
119		Hwy 10	SB		9
120		Miss.	Off Ramp		35
121		Southdown	Off Ramp		84
122		Winston	1		97
123		Winston	2		98
124		Ford Dr.	1		135
125		Ford Dr.	2		136
126	QUEUE OCCUPANCY	Cawthra	NB		515
127		Cawthra	SB		518
128		Hwy 10	NB		7
129		Hwy 10	SB		8
130		Miss.	NB		30
131		Miss.	SBE		33
132		Miss.	SBW		34
133		Southdown	S	30-s	70

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
134		Southdown	N	30-s	77
135		Winston	NB		93
136		Winston	SB		96
137		Ford Dr.	S		115
138		Ford Dr.	N		116
139	OCC	15A	Driving		36
140	OCC	15A	Centre		37
141	DOWNSTREAM VOLUME	22	Driv (1)		528
142		22	Cntr (2)		530
143		22	Pass (3)		532
144		21	1		522
145		21	2		524
146		21	3		526
147		20	1		508
148		20	2		510
149		20	3		512
150		19	1		502
151		19	2		504
152		19	3		506
153		18	1		11
154		18	2		13
155		18	3		15
156		17	1		17
157		17	2		19
158		17	3		21
159		16	1		23
160		16	2		25
161		16	3		27
162		15	Driv (1)		39
163		15	Cntr (2)		41
164		15	Pass (3)		43
165		14	Driv (1)		45
166		14	Cntr (2)		47
167		14	Pass (3)		49
168		13	Driv (1)		51
169		13	Cntr (2)		53
170		13	Pass (3)		55
171		12	1		57
172		12	2		59
173		12	3		61
174		11	1		63
175		11	2		65
176		11	3		67
177		10	1		72
178		10	2		74
179		10	3		76
180		9	1		79
181		9	2		81
182		9	3		83
183		8	1	30-s	86

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
184	DOWNSTREAM VOLUME	8	2	30-s	88
185		8	3		90
186		7	1		100
187		7	2		102
188		7	3		104
189		6	Driving		106
190		6	R. Centre		108
191		6	L. Centre		110
192		6	Passing		112
193		5	1		118
194		5	2		120
195		5	3		122
196		4	Driv (1)		124
197		4	Cntr (1)		126
198		4	Pass (3)		128
199		3	Driv (1)		130
200		3	Cntr (2)		132
201		3	Pass (3)		134
202		2	1		138
203		2	2		140
204		2	3		142
205		1	Driv (1)		144
206		1	Cntr (2)	↓	146
207	↓ ↓	1	Pass (3)	30-s	148
208	UPSTREAM VOLUME	22	Driv (1)	30-s	527
209		22	Cntr (2)		529
210		22	Pass (3)		531
211		21	1		521
212		21	2		523
213		21	3		525
214		20	1		507
215		20	2		509
216		20	3		511
217		19	1		501
218		19	2		503
219		19	3		505
220		18	1		10
221		18	2		12
222		18	3		14
223		17	1		16
224		17	2		18
225		17	3		20
226		16	1		22
227		16	2		24
228		16	3		26
229		15	Driv (1)		38
230		15	Cntr (2)		40
231		15	Pass (3)		42
232		14	Driv (1)		44
233		14	Cntr (2)		46
234	↓ ↓	14	Pass (3)	↓	48
235		13	1	30 s	50

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
236	UPSTREAM VOLUME	13	2	30-s	52
237		13	3		54
238		12	1		56
239		12	2		58
240		12	3		60
241		11	1		62
242		11	2		64
243		11	3		66
244		10	1		71
245		10	2		73
246		10	3		75
247		9	1		78
248		9	2		80
249		9	3		82
250		8	1		85
251		8	2		87
252		8	3		89
253		7	1		99
254		7	2		101
255		7	3		103
256		6	Driving		105
257		6	R. Centre		107
258		6	L. Centre		109
259		6	Passing		111
260		5	1		117
261		5	2		119
262		5	3		121
263		4	1		123
264		4	2		125
265		4	3		127
266		3	1		129
267		3	2		131
268		3	3		133
269		2	1		137
270		2	2		139
271		2	3		141
272		1	Driv (1)		143
273		1	Cntr (2)		145
274		1	Pass (3)	30-s	147
275	DOWNSTREAM OCCUPANCY	22	Driv (1)	30-s	528
276		22	Cntr (2)		530
277		22	Pass (3)		532
278		21	1		522
279		21	2		524
280		21	3		526
281		20	1		508
282		20	2		510
283		20	3	30-s	512

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
284	DOWNSTREAM OCCUPANCY	19	1	30-s	502
285		19	2		504
286		19	3		506
287		18	1		11
288		18	2		13
289		18	3		15
290		17	1		17
291		17	2		19
292		17	3		21
293		16	1		23
294		16	2		25
295		16	3		27
296		15	Driv (1)		39
297		15	Cntr (2)		41
298		15	Pass (3)		43
299		14	Driv (1)		45
300		14	Cntr (2)		47
301		14	Pass (3)		49
302		13	1		51
303		13	2		53
304		13	3		55
305		12	1		57
306		12	2		59
307		12	3		61
308		11	1		63
309		11	2		65
310		11	3		67
311		10	1		72
312		10	2		74
313		10	3		76
314		9	1		79
315		9	2		81
316		9	3		83
317		8	1		86
318		8	2		88
319		8	3		90
320		7	1		100
321		7	2		102
322		7	3		104
323		6	Driving		106
324		6	R. Centre		108
325		6	L. Centre		110
326		6	Passing		112
327		5	1		118
328		5	2		120
329		5	3	30-s	122

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
330	DOWNSTREAM OCCUPANCY	4	1	30-s	124
331		4	2		126
332		4	3		128
333		3	1		130
334		3	2		132
335		3	3		134
336		2	1		138
337		2	2		140
338		2	3		142
339		1	Driv (1)		144
340		1	Cntr (2)		146
341		1	Pass (3)	30-s	148
342	UPSTREAM OCCUPANCY	22	Driv (1)	30-s	527
343		22	Cntr (2)		529
344		22	Pass (3)		531
345		21	1		521
346		21	2		523
347		21	3		525
348		20	1		507
349		20	2		509
350		20	3		511
351		19	1		501
352		19	2		503
353		19	3		505
354		18	1		10
355		18	2		12
356		18	3		14
357		17	1		16
358		17	2		18
359		17	3		20
360		16	Driv (1)		22
361		16	Cntr (2)		24
362		16	Pass (3)		26
363		15	Driv (1)		38
364		15	Cntr (2)		40
365		15	Pass (3)		42
366		14	1		44
367		14	2		46
368		14	3		48
369		13	1		50
370		13	2		52
371		13	3		54
372		12	1		56
373		12	2		58
374		12	3		60
375		11	1		62
376		11	2		64
377		11	3	30-s	66

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
378	UPSTREAM OCCUPANCY	10	1	30-s	71
379		10	2		73
380		10	3		75
381		9	1		78
382		9	2		80
383		9	3		82
384		8	1		85
385		8	2		87
386		8	3		89
387		7	1		99
388		7	2		101
389		7	3		103
390		6	Driving		105
391		6	R. Center		107
392		6	L. Centre		109
393		6	Passing		111
394		5	Driv (1)		117
395		5	Cntr (2)		119
396		5	Pass (3)		121
397		4	Driv (1)		123
398		4	Cntr (2)		125
399		4	Pass (3)		127
400		3	1		129
401		3	2		131
402		3	3		133
403		2	1		137
404		2	2		139
405		2	3		141
406		1	Driv (1)		143
407		1	Cntr (2)		145
408		1	Pass (3)	30-s	147
409	SPEED (M/S)	22	Driv (1)	30-s	
410		22	Cntr (2)		
411		22	Pass (3)		
412		21	1		
413		21	2		
414		21	3		
415		20	1		
416		20	2		
417		20	3		
418		19	1		
419		19	2		
420		19	3		
421		18	1		
422		18	2		
423	SPEED (M/S)	18	3	30-s	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
424	SPEED (M/S)	17	1	30-s	
425		17	2		
426		17	3		
427		16	Driv		
428		16	Cntr		
429		16	Pass		
430		15	Driv (1)		
431		15	Cntr (2)		
432		15	Pass (3)		
433		14	1		
434		14	2		
435		14	3		
436		13	1		
437		13	2		
438		13	3		
439		12	1		
440		12	2		
441		12	3		
442		11	1		
443		11	1		
444		11	3		
445		10	1		
446		10	2		
447		10	3		
448		9	1		
449		9	2		
450		9	3		
451		8	1		
452		8	2		
453		8	3		
454		7	1		
455		7	2		
456		7	3		
457		6	Driving		
458		6	R. Centre		
459		6	L. Centre		
460		6	Passing		
461		5	Driv		
462		5	Cntr		
463		5	Pass		
464		4	1		
465		4	2		
466		4	3		
467		3	1		
468		3	2		
469	SPEED (M/S)	3	3	30-s	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
470	SPEED (M/S)	2	1	30-s	
471		2	2		
472		2	3		
473		1	Driv (1)		
474	↓ ↓	1	Cntr (2)	↓	
475		1	Pass (3)	30-s	
476	LONG VEHICLE COUNT	22	Driv (1)	30-s	
477		22	Cntr (2)		
478		22	Pass (3)		
479		21	1		
480		21	2		
481		21	3		
482		20	1		
483		20	2		
484		20	3		
485		19	1		
486		19	2		
487		19	3		
488		18	1		
489		18	2		
490		18	3		
491		17	1		
492		17	2		
493		17	3		
494		16	Driv (1)		
495		16	Cntr (2)		
496		16	Pass (3)		
497		15	1		
498		15	2		
499		15	3		
500		14	1		
501		14	2		
502		14	3		
503		13	1		
504		13	2		
505		13	3		
506		12	1		
507		12	2		
508		12	3		
509		11	1		
510		11	2		
511		11	3		
512		10	1		
513		10	2		
514		10	3		
515		9	1		
516	↓ ↓	9	2	↓	
517		9	3	30-s	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
518	LONG VEHICLE COUNT	8	1	30-s	
519		8	2		
520		8	3		
521		7	1		
522		7	2		
523		7	3		
524		6	Driving (1)		
525		6	R. Center		
526		6	L. Center		
527		6	Passing (3)		
528		5	Driv (1)		
529		5	Cntr (2)		
530		5	Pass (3)		
531		4	1		
532		4	2		
533		4	3		
534		3	1		
535		3	2		
536		3	3		
537		2	1		
538		2	2		
539		2	3		
540		1	Driv (1)		
541		1	Cntr (2)		
542		1	Pass (3)	30-s	
543	STATION OCCUPANCY	1		30-s	
544		2			
545		3			
546		4			
547		5			
548		6			
549		7			
550		8			
551		9			
552		10			
553		11			
554		12			
555		13			
556		14			
557		15			
558		16			
559		17			
560		18			
561		19			
562		20			
563		21			
564		22		30-s	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
565	STATION OCCUPANCY	1		PRECEEDING	30-s
566		2			
567		3			
568		4			
569		5			
570		6			
571		7			
572		8			
573		9			
574		10			
575		11			
576		12			
577		13			
578		14			
579		15			
580		16			
581		17			
582		18			
583		19			
584		20			
585	↓ ↓	21		↓	↓
586		22		PRECEEDING	30-s
587	STATION OCCUPANCY	1		1 Min	
588		2			
589		3			
590		4			
591		5			
592		6			
593		7			
594		8			
595		9			
596		10			
597		11			
598		12			
599		13			
600		14			
601		15			
602		16			
603		17			
604		18			
605		19			
606		20			
607	↓ ↓	21			
608		22			
609	STATION VOLUME	1			
610		2			
611	↓ ↓	3		↓	
612		4		1 Min	

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
613	STATION VOLUME	5		1 Min	
614		6			
615		7			
616		8			
617		9			
618		10			
619		11			
620		12			
621		13			
622		14			
623		15			
624		16			
625		17			
626		18			
627		19			
628		20			
629	↓ ↓	21		↓	
630		22		1 Min	
631	STATION VOLUME	1		30 s	
632		2			
633		3			
634		4			
635		5			
636		6			
637		7			
638		8			
639		9			
640		10			
641		11			
642		12			
643		13			
644		14			
645		15			
646		16			
647		17			
648		18			
649		19			
650		20			
651	↓ ↓	21			
652		22		30 s	
653	STATION VOLUME	1		PRECEEDING	30-s
654		2			
655		3			
656		4			
657		5			
658		6			
659	↓ ↓	7		↓	↓
660		8		PRECEEDING	30-s

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
661	STATION VOLUME	9		PRECEEDING	30-s
662		10			
663		11			
664		12			
665		13			
666		14			
667		15			
668		16			
669		17			
670		18			
671		19			
672		20			
673	↓ ↓	21		↓	↓
674		22		PRECEEDING	30-s
675	STATION SPEED	1		30-s	
676		2			
677		3			
678		4			
679		5			
680		6			
681		7			
682		8			
683		9			
684		10			
685		11			
686		12			
687		13			
688		14			
689		15			
690		16			
691		17			
692		18			
693		19			
694		20			
695	↓ ↓	21		↓	
696		22		30-s	
697	LANE STATUS	1	ALL		
698		2			
699		3			
700		4			
701		5			
702		6			
703		7			
704		8			
705		9			
706	↓ ↓	10	↓		
707		11	ALL		

Appendix 27 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector #
708	LANE STATUS	12	ALL		
709	↓	13	↓		
710	↓	14	↓		
711	↓	15	↓		
712	↓	16	↓		
713	↓	17	↓		
714	↓	18	↓		
715	↓	19	↓		
716	↓	20	↓		
717	↓	21			
718	↓	22	ALL		

Appendix 28/ Data-5M File Logical Record

Word#	Content	Station#	Lane#	Time Slice	Detector#
1	Hour				
2	Min			5 Min.	
3	s				
4	PASSAGE VOLUME	Cawthra	NB	5 Min.	513
5		Cawthra	SB		516
6		Hwy 10	NB		1
7		Hwy 10	SB		3
8		Miss.	NB		28
9		Miss.	SB		31
10		Southdown	On Ramp		68
11		Winston	NB		91
12		Winston	SB		94
13		Ford Dr.	On Ramp		113
14	DEMAND VOLUME	Cawthra	NB		514
15		Cawthra	SB		517
16		Hwy 10	NB		2
17		Hwy 10	SB		4
18		Miss.	NB		29
19		Miss.	SB		32
20		Southdown	On Ramp		69
21		Winston	NB		92
22		Winston	SB		95
23		Ford Dr.	On Ramp		114
24	EXIT VOLUME	Cawthra	1		519
25		Cawthra	2		520
26		Hwy 10	Service		5
27		Hwy 10	NB		6
28		Hwy 10	SB		9
29		Miss.	Off Ramp		35
30		Southdown	Off Ramp		84
31		Winston	1		97
32		Winston	2		98
33		Ford Dr.	1		135
34		Ford Dr.	2		136
35	QUEUE VOLUME	Cawthra	NB		515
36		Cawthra	SB		518
37		Hwy 10	NB		7
38		Hwy 10	SB		8
39		Miss.	NB		30
40		Miss.	SB E		33
41		Miss.	SB W		34
42		Southdown	S		70
43		Southdown	N		77
44		Winston	NB		93
45		Winston	SB		96
46		Ford Dr.	S		115
47		Ford Dr.	N	5 Min.	116

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
48	VOL	15A	Driv	5 Min.	36
49	VOL	15A	Pass		37
50	PASSAGE OCCUPANCY	Cawthra	NB		513
51		Cawthra	SB		516
52		Hwy 10	NB		1
53		Hwy 10	SB		3
54		Miss.	NB		28
55		Miss.	SB		31
56		Southdown	On Ramp		68
57		Winston	NB		91
58		Winston	SB		94
59		Ford Dr.	On Ramp		113
60	DEMAND OCCUPANCY	Cawthra	NB		514
61		Cawthra	SB		517
62		Hwy 10	NB		2
63		Hwy 10	SB		4
64		Miss.	NB		29
65		Miss.	SB		32
66		Southdown	On Ramp		69
67		Winston	NB		92
68		Winston	SB		95
69		Ford Dr.	On Ramp		114
70	EXIT OCCUPANCY	Cawthra	1		519
71		Cawthra	2		520
72		Hwy 10	Service		5
73		Hwy 10	NB		6
74		Hwy 10	SB		9
75		Miss.	Off Ramp		35
76		Southdown	Off Ramp		89
77		Winston	1		97
78		Winston	2		98
79		Ford Dr.	1		135
80		Ford Dr.	2		136
81	QUEUE OCCUPANCY	Cawthra	NB		515
82		Cawthra	SB		518
83		Hwy 10	NB		7
84		Hwy 10	SB		8
85		Miss.	NB		30
86		Miss.	SB E		33
87		Miss.	SB W		34
88		Southdown	S		70
89		Southdown	N		77
90		Winston	NB		93
91		Winston	SB		96
92		Ford Dr.	S		115
93		Ford Dr.	N	5 Min.	116

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
94	OCC.	15A	Driv	5 Min.	36
95	OCC.	15A	Pass		37
96	VOLUME	22	Driv		528
97		22	Cntr		530
98		21	Pass		532
99		21	Driv		522
100		21	Cntr		524
101		21	Pass		526
102	DOWNSTREAM VOLUME	20	Driv		508
103		20	Cntr		510
104		20	Pass		512
105		19	Driv		502
106		19	Cntr		504
107		19	Pass		506
108		18	Driv		11
109		18	Cntr		13
110		18	Pass		15
111		17	Driv		17
112		17	Cntr		19
113		17	Pass		21
114		16	Driv		23
115		16	Cntr		25
116		16	Pass		27
117		15	Driv		39
118		15	Cntr		41
119		15	Pass		43
120		14	Driv		45
121		14	Cntr		47
122		14	Pass		49
123		13	Driv		51
124		13	Cntr		53
125		13	Pass		55
126		12	Driv		57
127		12	Cntr		59
128		12	Pass		61
129	VOLUME	11	Driv		63
130		11	Cntr		65
131		11	Pass		67
132		10	Driv		72
133		10	Cntr		74
134	VOLUME	10	Pass		76
135	DOWNSTREAM VOLUME	9	Driv		79
136		9	Cntr		81
137		9	Pass		83
138		8	Driv		86
139		8	Cntr		88
140		8	Pass	5 Min.	90

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
141	DOWNSTREAM VOLUME	7	Driv	5 Min.	100
142		7	Cntr		102
143		7	Pass		104
144		6	Driv		106
145		6	R. Centre		108
146		6	L. Centre		110
147		6	Pass		112
148		5	Driv		118
149		5	Cntr		120
150		5	Pass		122
151		4	Driv		124
152		4	Cntr		126
153		4	Pass		128
154		3	Driv		130
155		3	Cntr		132
156		3	Pass		134
157		2	Driv		138
158		2	Cntr		140
159		2	Pass		142
160		1	Driv		144
161		1	Cntr		146
162		1	Pass		148
163	UPSTREAM VOLUME	22	Driv		527
164		22	Cntr		529
165		22	Pass		531
166		21	Driv (1)		521
167		21	Cntr (2)		523
168		21	Pass (3)		525
169		20	Driv (1)		507
170		20	Cntr (2)		509
171		20	Pass (3)		511
172		19	Driv (1)		501
173		19	Cntr (2)		503
174		19	Pass (3)		505
175		18	Driv (1)		10
176		18	Cntr (2)		12
177		18	Pass (3)		14
178		17	Driv (1)		16
179		17	Cntr (2)		18
180		17	Pass (3)		20
181		16	Driv (1)		22
182		16	Cntr (2)		24
183		16	Pass (3)		26
184		15	Driv (1)		38
185		15	Cntr (2)		40
186		15	Pass (3)	5 Min.	42

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
187	UPSTREAM VOLUME	14	Driv (1)	5 Min.	44
188		14	Cntr (2)		46
189		14	Pass (3)		48
190		13	Driv (1)		50
191		13	Cntr (2)		52
192		13	Pass (3)		54
193		12	Driv (1)		56
194		12	Cntr (2)		58
195		12	Pass (3)		60
196		11	Driv (1)		62
197		11	Cntr (2)		64
198		11	Pass (3)		66
199		10	Driv (1)		71
200		10	Cntr (2)		73
201		10	Pass (3)		75
202		9	1		78
203		9	2		80
204		9	3		82
205		8	1		85
206		8	2		87
207		8	3		89
208		7	1		99
209		7	2		101
210		7	3		103
211		6	Driv		105
212		6	R. Centre		107
213		6	L. Centre		109
214		6	Pass		111
215		5	1		117
216		5	2		119
217		5	3		121
218		4	1		123
219		4	2		125
220		4	3		127
221		3	1		129
222		3	2		131
223		3	3		133
224		2	1		137
225		2	2		139
226		2	3		141
227		1	1		143
228		1	2		145
229		1	3		147
230	DOWNSTREAM OCCUPANCY	22	Driv (1)		528
231		22	Cntr (2)		530
232		22	Pass (3)	5 Min.	532

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
233	DOWNSTREAM OCCUPANCY	21	1	5 Min.	522
234		21	2		524
235		21	3		526
236		20	1		508
237		20	2		510
238		20	3		512
239		19	1		502
240		19	2		504
241		19	3		506
242		18	1		11
243		18	2		13
244		18	3		15
245		17	1		17
246		17	2		19
247		17	3		21
248		16	1		23
249		16	2		25
250		16	3		27
251		15	1		39
252		15	2		41
253		15	3		43
254		14	1		45
255		14	2		47
256		14	3		49
257		13	1		51
258		13	2		53
259		13	3		55
260		12	1		57
261		12	2		59
262		12	3		61
263		11	1		63
264		11	2		65
265		11	Pass (3)		67
266		10	Driv (1)		72
267		10	Cntr (2)		74
268		10	Pass (3)		76
269		9	1		79
270		9	2		81
271		9	3		83
272		8	1		86
273		8	2		88
274		8	3		90
275		7	1		100
276		7	2		102
277		7	3	5 Min.	104

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
278	DOWNSTREAM OCCUPANCY	6	Driv	5 Min.	106
279		6	R. Centre		108
280		6	L. Centre		110
281		6	Pass		112
282		5	1		118
283		5	2		120
284		5	3		122
285		4	1		124
286		4	2		126
287		4	3		128
288		3	1		130
289		3	2		132
290		3	3		134
291		2	1		138
292		2	2		140
293		2	3		142
294		1	Driv (1)		144
295		1	Cntr (2)		146
296		1	Pass (3)		148
297	UPSTREAM OCCUPANCY	22	Driv (1)		527
298		22	Cntr (2)		529
299		22	Pass (3)		531
300		21	Driv (1)		521
301		21	Cntr (2)		523
302		21	Pass (3)		525
303		20	1		507
304		20	2		509
305		20	3		511
306		19	1		501
307		19	2		503
308		19	3		505
309		18	1		10
310		18	2		12
311		18	3		14
312		17	1		16
313		17	2		18
314		17	3		20
315		16	1		22
316		16	2		24
317		16	3		26
318		15	1		38
319		15	2		40
320		15	3		42
321		14	1		44
322		14	2		46
323		14	3	5 Min.	48

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
324	UPSTREAM OCCUPANCY	13	1	5 Min.	50
325		13	2		52
326		13	3		54
327		12	Driv (1)		56
328		12	Cntr (2)		58
329		12	Pass (3)		60
330		11	Driv (1)		62
331		11	Cntr (2)		64
332		11	Pass (3)		66
333		10	Driv (1)		71
334		10	Cntr (2)		73
335		10	Pass (3)		75
336		9	1		78
337		9	2		80
338		9	3		82
339		8	1		85
340		8	2		87
341		8	3		89
342		7	1		99
343		7	2		101
344		7	3		103
345		6	Driv		105
346		6	R. Centre		107
347		6	L. Centre		109
348		6	Pass		111
349		5	1		117
350		5	2		119
351		5	3		121
352		4	1		123
353		4	2		125
354		4	3		127
355		3	1		129
356		3	2		131
357		3	3		133
358		2	1		137
359		2	2		139
360		2	3		141
361		1	Driv (1)		143
362		1	Cntr (2)		145
363		1	Pass (3)		147
364	LONG VEHICLE COUNT	22	Driv (1)		
365		22	Cntr (2)		
366		22	Pass (3)		
367		21	1		
368		21	2		
369		21	3	5-Min.	

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
370	LONG VEHICLE COUNT	20	1	5 Min.	
371		20	2		
372		20	3		
373		19	1		
374		19	2		
375		19	3		
376		18	1		
377		18	2		
378		18	3		
379		17	1		
380		17	2		
381		17	3		
382		16	1		
383		16	2		
384		16	3		
385		15	1		
386		15	2		
387		15	3		
388		14	1		
389		14	2		
390		14	3		
391		13	1		
392		13	2		
393		13	3		
394		12	1		
395		12	2		
396		12	3		
397		11	Driv (1)		
398		11	Cntr (2)		
399		11	Pass (3)		
400		10	1		
401		10	2		
402		10	3		
403		9	1		
404		9	2		
405		9	3		
406		8	1		
407		8	2		
408		8	3		
409		7	1		
410		7	2		
411		7	3		
412		6	Driv		
413		6	R. Centre		
414		6	L. Centre		
415		6	Pass	5 Min.	

Appendix 28 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
416	LONG VEHICLE COUNT	5	1	5 Min.	
417		5	2		
418		5	3		
419		4	1		
420		4	2		
421		4	3		
422		3	1		
423		3	2		
424		3	3		
425		2	1		
426		2	2		
427		2	3		
428		1	Driv (1)		
429		1	Cntr (2)		
430		1	Pass (3)	5 Min.	

Appendix 29/ QEWDYN File Logical Record

Word#	Content	Station#	Lane#	Time Slice	Detector#
1	Year			↑	
2	Month			1 Min.	
3	Day			Interval	
4	Hour				
5	min				
6	s			↓	
7	MODCYC	Cawthra	NB	Latest 30 s	
8	Cycle Length	Cawthra	NB		
9	Queue Wait	Cawthra	NB	↓	
10	Queue Occupancy	Cawthra	NB	Smoothed 30 s	
11	Reserved	Cawthra	NB		
12	MODCYC	Cawthra	SB	Latest 30 s	
13	Cycle Length	Cawthra	SB		
14	Queue Wait	Cawthra	SB	↓	
15	Queue Occupancy	Cawthra	SB	Smoothed 30 s	
16	Reserved	Cawthra	SB		
17	MODCYC	Hwy 10	NB	Latest 30 s	
18	Cycle Length	Hwy 10	NB		
19	Queue Wait	Hwy 10	NB	↓	
20	Queue Occupancy	Hwy 10	NB	Smoothed 30 s	
21	Reserved	Hwy 10	NB		
22	MODCYC	Hwy 10	SB	Latest 30 s	
23	Cycle Length	Hwy 10	SB		
24	Queue Wait	Hwy 10	SB	↓	
25	Queue Occupancy	Hwy 10	SB	Smoothed 30 s	
26	Reserved	Hwy 10	SB		
27	MODCYC	Miss.	NB	Latest 30 s	
28	Cycle Length	Miss.	NB		
29	Queue Wait	Miss.	NB	↓	
30	Queue Occupancy	Miss.	NB	Smoothed 30 s	
31	Reserved	Miss.	NB		
32	MODCYC	Miss.	SB	Latest 30 s	
33	Cycle Length	Miss.	SB		
34	Queue Wait	Miss.	SB	↓	
35	Queue Occupancy	Miss.	SB	Smoothed 30 s	
36	Reserved	Miss.	SB		
37	MODCYC	Southdown	--	Latest 30 s	
38	Cycle Length	Southdown	--		
39	Queue Wait	Southdown	--	↓	
40	Queue Occupancy	Southdown	--	Smoothed 30 s	
41	Reserved	Southdown	--		
42	MODCYC	Winston	NB	Latest 30 s	
43	Cycle Length	Winston	NB		
44	Queue Wait	Winston	NB	↓	
45	Queue Occupancy	Winston	NB	Smoothed 30 s	
46	Reserved	Winston	NB		

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
47	MODCYC	Winston	SB	Latest 30 s	
48	Cycle Length	Winston	SB	↓	
49	Queue Wait	Winston	SB	↓	
50	Queue Occupancy	Winston	SB	Smoothed 30 s	
51	Reserved	Winston	SB		
52	MODCYC	Ford Dr.	--	Latest 30 s	
53	Cycle Length	Ford Dr.	--	↓	
54	Queue Wait	Ford Dr.	--	↓	
55	Queue Occupancy	Ford Dr.	--	Smoothed 30 s	
56	Reserved	Ford Dr.	--		
57	Capacity	22	--	1 min.	
58	Speed (km/h)	22	--	Latest 30 s	
59	Volume	22	--	1 min.	
60	Occupancy	22	--	1 min.	
61	Incident	22	--	Latest 30 s	
62	Capacity	21	--	1 min.	
63	Speed (km/h)	21	--	Latest 30 s	
64	Volume	21	--	1 min.	
65	Occupancy	21	--	1 min.	
66	Incident	21	--	Latest 30 s	
67	Capacity	20	--	1 min.	
68	Speed (km/h)	20	--	Latest 30 s	
69	Volume	20	--	1 min.	
70	Occupancy	20	--	1 min.	
71	Incident	20	--	Latest 30 s	
72	Capacity	19	--	1 min.	
73	Speed (km/h)	19	--	Latest 30 s	
74	Volume	19	--	1 min.	
75	Occupancy	19	--	1 min.	
76	Incident	19	--	Latest 30 s	
77	Capacity	18	--	1 min.	
78	Speed (km/h)	18	--	Latest 30 s	
79	Volume	18	--	1 min.	
80	Occupancy	18	--	1 min.	
81	Incident	18	--	Latest 30 s	
82	Capacity	17	--	1 min.	
83	Speed (km/h)	17	--	Latest 30 s	
84	Volume	17	--	1 min.	
85	Occupancy	17	--	1 min.	
86	Incident	17	--	Latest 30 s	
87	Capacity	16	--	1 min.	
88	Speed (km/h)	16	--	Latest 30 s	
89	Volume	16	--	1 min.	
90	Occupancy	16	--	1 min.	
91	Incident	16	--	Latest 30 s	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
92	Capacity	15	--	1 min.	
93	Speed (km/h)	15	--	Latest 30 s	
94	Volume	15	--	1 min.	
95	Occupancy	15	--	1 min.	
96	Incident	15	--	Latest 30 s	
97	Capacity	14	--	1 min.	
98	Speed (km/h)	14	--	Latest 30 s	
99	Volume	14	--	1 min.	
100	Occupancy	14	--	1 min.	
101	Incident	14	--	Latest 30 s	
102	Capacity	13	--	1 min.	
103	Speed (km/h)	13	--	Latest 30 s	
104	Volume	13	--	1 min.	
105	Occupancy	13	--	1 min.	
106	Incident	13	--	Latest 30 s	
107	Capacity	12	--	1 min.	
108	Speed (km/h)	12	--	Latest 30 s	
109	Volume	12	--	1 min.	
110	Occupancy	12	--	1 min.	
111	Incident	12	--	Latest 30 s	
112	Capacity	11	--	1 min.	
113	Speed (km/h)	11	--	Latest 30 s	
114	Volume	11	--	1 min.	
115	Occupancy	11	--	1 min.	
116	Incident	11	--	Latest 30 s	
117	Capacity	10	--	1 min.	
118	Speed (km/h)	10	--	Latest 30 s	
119	Volume	10	--	1 min.	
120	Occupancy	10	--	1 min.	
121	Incident	10	--	Latest 30 s	
122	Capacity	9	--	1 min.	
123	Speed (km/h)	9	--	Latest 30 s	
124	Volume	9	--	1 min.	
125	Occupancy	9	--	1 min.	
126	Incident	9	--	Latest 30 s	
127	Capacity	8	--	1 min.	
128	Speed (km/h)	8	--	Latest 30 s	
129	Volume	8	--	1 min.	
130	Occupancy	8	--	1 min.	
131	Incident	8	--	Latest 30 s	
132	Capacity	7	--	1 min.	
133	Speed (km/h)	7	--	Latest 30 s	
134	Volume	7	--	1 min.	
135	Occupancy	7	--	1 min.	
136	Incident	7	--	Latest 30 s	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
137	Capacity	6	--	1 min.	
138	Speed (km/h)	6	--	Latest 30 s	
139	Volume	6	--	1 min.	
140	Occupancy	6	--	1 min.	
141	Incident	6	--	Latest 30 s	
142	Capacity	5	--	1 min.	
143	Speed (km/h)	5	--	Latest 30 s	
144	Volume	5	--	1 min.	
145	Occupancy	5	--	1 min.	
146	Incident	5	--	Latest 30 s	
147	Capacity	4	--	1 min.	
148	Speed (km/h)	4	--	Latest 30 s	
149	Volume	4	--	1 min.	
150	Occupancy	4	--	1 min.	
151	Incident	4	--	Latest 30 s	
152	Capacity	3	--	1 min.	
153	Speed (km/h)	3	--	Latest 30 s	
154	Volume	3	--	1 min.	
155	Occupancy	3	--	1 min.	
156	Incident	3	--	Latest 30 s	
157	Capacity	2	--	1 min.	
158	Speed (km/h)	2	--	Latest 30 s	
159	Volume	2	--	1 min.	
160	Occupancy	2	--	1 min.	
161	Incident	2	--	Latest 30 s	
162	Capacity	1	--	1 min.	
163	Speed (km/h)	1	--	Latest 30 s	
164	Volume	1	--	1 min.	
165	Occupancy	1	--	1 min.	
166	Incident	1	--	Latest 30 s	
167	Volume	15A	--	1 min.	
168	Occupancy	15A	--	↓	
169	Volume Passage	Cawthra	NB		513
170	Occupancy	Cawthra	NB		513
171	Volume Demand	Cawthra	NB		514
172	Occupancy	Cawthra	NB		514
173	Volume Queue	Cawthra	NB		515
174	Occupancy	Cawthra	NB		515
175	Volume Passage	Cawthra	SB		516
176	Occupancy	Cawthra	SB		516
177	Volume Demand	Cawthra	SB		517
178	Occupancy	Cawthra	SB		517
179	Volume Queue	Cawthra	SB		518
180	Occupancy	Cawthra	SB		518
181	Volume Exit	Cawthra	Exit		519
182	Occupancy	Cawthra	Exit	1 min.	519

Appendix 29 (Continued)

Word#	Content		Station#	Lane#	Time Slice	Detector#
183	Volume	Exit	Cawthra	Exit	1 min.	520
184	Occupancy		Cawthra	Exit		520
185	Volume	Passage	Hwy 10	NB		1
186	Occupancy		Hwy 10	NB		1
187	Volume	Demand	Hwy 10	NB		2
188	Occupancy		Hwy 10	NB		2
189	Volume	Queue	Hwy 10	NB		7
190	Occupancy		Hwy 10	NB		7
191	Volume	Exit	Hwy 10	Service		5
192	Occupancy		Hwy 10	Service		5
193	Volume	Exit	Hwy 10	NB		6
194	Occupancy		Hwy 10	NB		6
195	Volume	Passage	Hwy 10	SB		3
196	Occupancy		Hwy 10	SB		3
197	Volume	Demand	Hwy 10	SB		4
198	Occupancy		Hwy 10	SB		4
199	Volume	Queue	Hwy 10	SB		8
200	Occupancy		Hwy 10	SB		8
201	Volume	Exit	Hwy 10	SB		9
202	Occupancy		Hwy 10	SB		9
203	Volume	Passage	Miss.	NB		28
204	Occupancy		Miss.	NB		28
205	Volume	Demand	Miss.	NB		29
206	Occupancy		Miss.	NB		29
207	Volume	Queue	Miss.	NB		30
208	Occupancy		Miss.	NB		30
209	Volume	Passage	Miss.	SB		31
210	Occupancy		Miss.	SB		31
211	Volume	Demand	Miss.	SB		32
212	Occupancy		Miss.	SB		32
213	Volume	Queue	Miss.	SB E		33
214	Occupancy		Miss.	SB E		33
215	Volume	Queue	Miss.	SB W		34
216	Occupancy		Miss.	SB W		34
217	Volume	Exit	Miss.	Off Ramp		35
218	Occupancy		Miss.	Off Ramp		35
219	Volume	Passage	Southdown	On Ramp		68
220	Occupancy		Southdown	On Ramp		68
221	Volume	Demand	Southdown	On Ramp		69
222	Occupancy		Southdown	On Ramp		69
223	Volume	Queue	Southdown	SB		70
224	Occupancy		Southdown	SB		70
225	Volume	Queue	Southdown	NB		77
226	Occupancy		Southdown	NB		77
227	Volume	Exit	Southdown	Off Ramp		84
228	Occupancy		Southdown	Off Ramp	1 min.	84

Appendix 29 (Continued)

Word#	Content		Station#	Lane#	Time Slice	Detector#
229	Volume	Passage	Winston	NB	1 min.	91
230	Occupancy		Winston	NB		91
231	Volume	Demand	Winston	NB		92
232	Occupancy		Winston	NB		92
233	Volume	Queue	Winston	NB		93
234	Occupancy		Winston	NB		93
235	Volume	Passage	Winston	SB		94
236	Occupancy		Winston	SB		94
237	Volume	Demand	Winston	SB		95
238	Occupancy		Winston	SB		95
239	Volume	Queue	Winston	SB		96
240	Occupancy		Winston	SB		96
241	Volume	Exit	Winston	1		97
242	Occupancy		Winston	1		97
243	Volume	Exit	Winston	2		98
244	Occupancy		Winston	2		98
245	Volume	Passage	Ford Dr.	On Ramp		113
246	Occupancy		Ford Dr.	On Ramp		113
247	Volume	Demand	Ford Dr.	On Ramp		114
248	Occupancy		Ford Dr.	On Ramp		114
249	Volume	Queue	Ford Dr.	SB On		115
250	Occupancy		Ford Dr.	Ramp		115
251	Volume	Queue	Ford Dr.	NB On		116
252	Occupancy		Ford Dr.	Ramp		116
253	Volume	Exit	Ford Dr.	1		135
254	Occupancy		Ford Dr.	1		135
255	Volume	Exit	Ford Dr.	2		136
256	Occupancy		Ford Dr.	2		136
257	Volume		22	Driv		
258	Occupancy		22	Driv		
259	Speed		22	Driv		
260	Volume		22	Centre		
261	Occupancy		22	Centre		
262	Speed		22	Centre		
263	Volume		22	Pass		
264	Occupancy		22	Pass		
265	Speed		22	Pass		
266	Volume		21	Driv		
267	Occupancy		21	Driv		
268	Speed		21	Driv		
269	Volume		21	Centre		
270	Occupancy		21	Centre		
271	Speed		21	Centre		
272	Volume		21	Pass		
273	Occupancy		21	Pass		
274	Speed		21	Pass	1 min.	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
275	Volume	20	Driv	1 min.	
276	Occupancy	20	Driv		
277	Speed	20	Driv		
278	Volume	20	Centre		
279	Occupancy	20	Centre		
280	Speed	20	Centre		
281	Volume	20	Pass		
282	Occupancy	20	Pass		
283	Speed	20	Pass		
284	Volume	19	Driv		
285	Occupancy	19	Driv		
286	Speed	19	Driv		
287	Volume	19	Centre		
288	Occupancy	19	Centre		
289	Speed	19	Centre		
290	Volume	19	Pass		
291	Occupancy	19	Pass		
292	Speed	19	Pass		
293	Volume	18	Driv		
294	Occupancy	18	Driv		
295	Speed	18	Driv		
296	Volume	18	Centre		
297	Occupancy	18	Centre		
298	Speed	18	Centre		
299	Volume	18	Pass		
300	Occupancy	18	Pass		
301	Speed	18	Pass		
302	Volume	17	Driv		
303	Occupancy	17	Driv		
304	Speed	17	Driv		
305	Volume	17	Centre		
306	Occupancy	17	Centre		
307	Speed	17	Centre		
308	Volume	17	Pass		
309	Occupancy	17	Pass		
310	Speed	17	Pass		
311	Volume	16	Driv		
312	Occupancy	16	Driv		
313	Speed	16	Driv		
314	Volume	16	Centre		
315	Occupancy	16	Centre		
316	Speed	16	Centre		
317	Volume	16	Pass		
318	Occupancy	16	Pass		
319	Speed	16	Pass	1 min.	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
320	Volume	15	Driv	1 min.	
321	Occupancy	15	Driv		
322	Speed	15	Driv		
323	Volume	15	Centre		
324	Occupancy	15	Centre		
325	Speed	15	Centre		
326	Volume	15	Pass		
327	Occupancy	15	Pass		
328	Speed	15	Pass		
329	Volume	14	Driv		
330	Occupancy	14	Driv		
331	Speed	14	Driv		
332	Volume	14	Centre		
333	Occupancy	14	Centre		
334	Speed	14	Centre		
335	Volume	14	Pass		
336	Occupancy	14	Pass		
337	Speed	14	Pass		
338	Volume	13	Driv		
339	Occupancy	13	Driv		
340	Speed	13	Driv		
341	Volume	13	Centre		
342	Occupancy	13	Centre		
343	Speed	13	Centre		
344	Volume	13	Pass		
345	Occupancy	13	Pass		
346	Speed	13	Pass		
347	Volume	12	Driv		
348	Occupancy	12	Driv		
349	Speed	12	Driv		
350	Volume	12	Centre		
351	Occupancy	12	Centre		
352	Speed	12	Centre		
353	Volume	12	Pass		
354	Occupancy	12	Pass		
355	Speed	12	Pass		
356	Volume	11	Driv		
357	Occupancy	11	Driv		
358	Speed	11	Driv		
359	Volume	11	Centre		
360	Occupancy	11	Centre		
361	Speed	11	Centre		
362	Volume	11	Pass		
363	Occupancy	11	Pass		
364	Speed	11	Pass	1 min.	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
365	Volume	10	Driv	1 min.	
366	Occupancy	10	Driv		
367	Speed	10	Driv		
368	Volume	10	Centre		
369	Occupancy	10	Centre		
370	Speed	10	Centre		
371	Volume	10	Pass		
372	Occupancy	10	Pass		
373	Speed	10	Pass		
374	Volume	9	Driv		
375	Occupancy	9	Driv		
376	Speed	9	Driv		
377	Volume	9	Centre		
378	Occupancy	9	Centre		
379	Speed	9	Centre		
380	Volume	9	Pass		
381	Occupancy	9	Pass		
382	Speed	9	Pass		
383	Volume	8	Driv		
384	Occupancy	8	Driv		
385	Speed	8	Driv		
386	Volume	8	Centre		
387	Occupancy	8	Centre		
388	Speed	8	Centre		
389	Volume	8	Pass		
390	Occupancy	8	Pass		
391	Speed	8	Pass		
392	Volume	7	Driv		
393	Occupancy	7	Driv		
394	Speed	7	Driv		
395	Volume	7	Centre		
396	Occupancy	7	Centre		
397	Speed	7	Centre		
398	Volume	7	Pass		
399	Occupancy	7	Pass		
400	Speed	7	Pass		
401	Volume	6	Driv		
402	Occupancy	6	Driv		
403	Speed	6	Driv		
404	Volume	6	R. Centre		
405	Occupancy	6	R. Centre		
406	Speed	6	R. Centre		
407	Volume	6	L. Centre		
408	Occupancy	6	L. Centre		
409	Speed	6	L. Centre	1 min.	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
410	Volume	6	Pass	1 min.	
411	Occupancy	6	Pass		
412	Speed	6	Pass		
413	Volume	5	Driv		
414	Occupancy	5	Driv		
415	Speed	5	Driv		
416	Volume	5	Centre		
417	Occupancy	5	Centre		
418	Speed	5	Centre		
419	Volume	5	Pass		
420	Occupancy	5	Pass		
421	Speed	5	Pass		
422	Volume	4	Driv		
423	Occupancy	4	Driv		
424	Speed	4	Driv		
425	Volume	4	Centre		
426	Occupancy	4	Centre		
427	Speed	4	Centre		
428	Volume	4	Pass		
429	Occupancy	4	Pass		
430	Speed	4	Pass		
431	Volume	3	Driv		
432	Occupancy	3	Driv		
433	Speed	3	Driv		
434	Volume	3	Centre		
435	Occupancy	3	Centre		
436	Speed	3	Centre		
437	Volume	3	Pass		
438	Occupancy	3	Pass		
439	Speed	3	Pass		
440	Volume	2	Driv		
441	Occupancy	2	Driv		
442	Speed	2	Driv		
443	Volume	2	Centre		
444	Occupancy	2	Centre		
445	Speed	2	Centre		
446	Volume	2	Pass		
447	Occupancy	2	Pass		
448	Speed	2	Pass		
449	Volume	1	Driv		
450	Occupancy	1	Driv		
451	Speed	1	Driv		
452	Volume	1	Centre		
453	Occupancy	1	Centre		
454	Speed	1	Centre		
455	Volume	1	Pass		
456	Occupancy	1	Pass		
457	Speed	1	Pass	1 min.	

Appendix 29 (Continued)

Word#	Content	Station#	Lane#	Time Slice	Detector#
458	Speed	Lakeshore/ Highway 2	W/B Passing	1 min.	
459	Volume				
460	Occupancy				
461	Speed		W/B Driving		
462	Volume				
463	Occupancy				
464	Speed		E/B Passing		
465	Volume				
466	Occupancy				
467	Speed		E/B Driving		
468	Volume				
469	Occupancy				
470	Speed	Dundas/ Highway 5	W/B Passing		
471	Volume				
472	Occupancy				
473	Speed		W/B Driving		
474	Volume				
475	Occupancy				
476	Speed		E/B Passing		
477	Volume				
478	Occupancy				
479	Speed		E/B Driving		
480	Volume				
481	Occupancy				

Appendix 30/INCLOG File Logical Record Format

Word#	Content	
1	Year	Date of Incident
2	Month	
3	Day	
4	Station No.	Incident location
5	Lane No.	
6	Hour	Incident starting time
7	Minute	
8	Second	
9	Hour	Incident ending time
10	Minute	
11	Second	
12	Incident duration	Incident effect and status
13	Maximum mainline queue length	
14	Downstream volume	
15	Maximum capacity reduction at incident site	
16	Travel time	
17	True or False	
18	Station No.	Incident location as reported by operator
19	Lane No.	
20	Hour	Incident starting time as reported by operator
21	Minute	
22	Seconds	
23	Hour	Incident ending time as reported by operator
24	Minute	
25	Second	

Appendix 31/ALMREC File Logical Record Format

Word#	Content	
1	Hour	Time of day
2	Minute	
3	Second	
4	Device identification	Identify device type, depending on device type, the meaning of Word No. 5 to 13 is modified
5	Device number	
6	Device location	
7	Device location	
8	Device location	
9	Error Code	
10	Bit position	
11	Type of data	
12	Value	
13	Program identification	
14	Checking period	

Footnote:

1. Device identification word

    1 .... Detector
    2 .... 710 ramp controller
    3 .... 170 ramp controller
    4 .... Ramp signal light
    5 .... Data modem
    6 .... CMS controller or sign

2. If device identification word equals to 1 (i.e., detector), then for a double loop detector (as identified by the device number).

    ALMREC (6) = Station number
    ALMREC (7) = Lane number (starting from driving lane)
    ALMREC (8) = 1 ... for downstream detector
    2 ... for upstream detector

Appendix 31 (Continued)

ALMREC (9) = 1 ... for occupancy
2 ... for volume
ALMREC (11) = 1 ... occupancy type
2 ... volume type

for a single loop detector
ALMREC (6) = Word No. + 1000
ALMREC (7) = 1 ... for passage detector
2 ... for demand detector
3 ... for exit detector
4 ... for queue detector
5 ... for mainline single loop detector
ALMREC (9) = 1 ... for occupancy
2 ... for volume
ALMREC (11) = 1 ... occupancy type
2 ... volume type

3. If device identification word equal to 2 or 3 (ramp controller), then
ALMREC (6) = ramp number
ALMREC (10) = bit position that is wrong

4. If the device identification word equal to 4 (ramp signal light), then
ALMREC (6) = ramp number
ALMREC (9) = 0 ... No lights were on during past 30 s
2 ... Only green light was on during past 30 s
3 ... Only amber light was on during past 30 s
4 ... Only red light was on during past 30 s
5 ... Only green and amber lights were on during past 30 s
6 ... Only green and red lights were on during past 30 s
7 ... Only amber and red lights were on during past 30 s
9 ... Green, amber, and red lights were on
ALMREC (11) = 1 ... In permanent green cycle
2 ... Normal metering cycle

Appendix 31 (Continued)

5. If device identification word equal to 5 (data modem), then
ALMREC (5) = octal modem address
(6) = modem number
ALMREC (9) = 1 MODEMT (4 ~ 76) 0
2 MODEMH > 0
3 BOTH
ALMREC (12) = MODEMH Value
ALMREC (13) = 3
ALMREC (14) = 600 s

6. If the device identification word equal to 6 (CMS), then
ALMREC (6) = first CMS

7. Program identification word
1 ALM30S
2 ALM5M
3 AMODM

Appendix 32/XMTTAB File Logical Record Format

Word#	Content
1	Year
2	Month
3	Day
4	Hour
5	Minute
6	Second
7	Left byte : binary zero; Right byte : modem address
8	Left byte : SOH; Right byte : BEL
9	2 message characters First line message
10	" " " " "
11	" " " " "
12	" " " " "
13	" " " " "
14	" " " " "
15	" " " " "
16	" " " " "
17	" " " " "
18	" " " " "
19	" " " " "
20	Left byte : CR; Right byte : LF
21	2 message characters Second line message
22	" " " " "
23	" " " " "
24	" " " " "
25	" " " " "
26	" " " " "
27	" " " " "
28	" " " " "
29	" " " " "
30	" " " " "
31	" " " " "
32	Left byte : EXT; Right byte : NUL

Appendix 32 (Continued)

Word#	Content
33	Unused
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"
46	"
47	"
48	"
49	"
50	"
51	"

Appendix 33 / RESDEV File Logical Record Format

Word#	Content	
1	Year	Date
2	Month	
3	Date	
4	Hour	Time of day
5	Minute	
6	Second	
7	Station No.	Incident Location
8	Lane No.	
9	Incident Occupancy Threshold Value	Various incident threshold value
10	Incident Volume Threshold Value	
11	Incident Speed Threshold Value	
12	Occupancy Vote weighting factor	Weighting factor to be used in weightingVTLU results
13	Volume vote weighting factor	
14	Speed vote weighting factor	
15	Occupancy CLDU time slice T vote	Voting result from occupancy CLDU
16	Occupancy CLDU time slice T-1 vote	
17	Occupancy CLDU time slice T-2 vote	
18	Result of majority decision rule based on occupancy VTLU	Result from occupancy VLTU
19	Volume CLDU time slice T vote	Voting result from volume CLDU
20	Volume CLDU time slice T-1 vote	
21	Volume CLDU time slice T-2 vote	
22	Result of majority decision rule based on volume VTLU	Result from volume VTLU

Appendix 33 (Continued)

Word #	Content
23	Speed CLDU time slice T vote ..... Voting result from speed CLDU
24	Speed CLDU time slice T-1 vote
25	Speed CLDU time slice T-2 vote
26	Result of majority decision ...... Result from speed VTLU rule based on speed VTLU

Appendix 34/ WEATHER File Logical Record Format

Word#	Content
1	Year .... Date
2	Month
3	Day
4	Hour .... Time
5	Minute
6	Second
7	W ; : .... Weather condition
8	ASCII character
9	"
10	"
11	"
12	"
13	"
14	"
15	"
16	"
17	P ; : .... Pavement condition
18	ASCII character
19	"
20	"
21	"
22	"
23	"
24	"
25	"
26	"
27	V ; ; .... Visibility condition
28	ASCII character
29	"
30	"
31	"
32	"

Appendix 34 (Continued)

Word#	Content
33	ASCII characters
34	"
35	"
36	"

Appendix 35/ First Physical Record Format of Data File

DATA30 - First physical record format

Word#	Content	
1	D ; A	File name
2	T ; A	
3	3 ; 0	
4	SP ; SP	
5	Year	Date
6	Month	
7	Day	
8	128	Physical record length (byte)
9	12	Number of physical record per logical record
10	718	Logical record length (word)
11	KL30S	Number of physical record
12	Reserved	
13	"	
14	"	
15	"	
16	"	
17	"	
18	"	
19	"	
20	"	
21	"	
22	"	
23	"	
24	"	
25	"	
26	"	
27	"	
29	"	
30	"	
31	"	
32	"	

Appendix 35 (Continued)

Word#	Content
33	Reserved
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"
46	"
47	"
48	"
49	"
50	"
51	"
52	"
53	"
54	"
55	"
56	"
57	"
58	"
59	"
60	"
61	"
62	"
63	"
64	"

Appendix 35 (Continued)

Data5M - First physical record format

Word#	Content
1	D ; A .... File name
2	T ; A
3	5 ; M
4	SP ; SP
5	Year .... Date
6	Month
7	Day
8	128 .... Physical record length (byte)
9	7 .... Number of physical record per logical record
10	430 .... Logical record length (word)
11	K5M .... Number of physical record
12	Reserved
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"
25	"
26	"
27	"
29	"
30	"
31	"
32	"

Appendix 35 (Continued)

Word#	Content
33	Reserved
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"
46	"
47	"
48	"
49	"
50	"
51	"
52	"
53	"
54	"
55	"
56	"
57	"
58	"
59	"
60	"
61	"
62	"
63	"
64	"

Appendix 35 (Continued)

QEWDYN - First physical record format

Word#	Content	
1	Q ; E	File name
2	W ; D	
3	Y ; N	
4	SP ; SP	
5	Year	Date
6	Month	
7	Day	
8	128	Physical record length (byte)
9	8	Number of physical record per logical record
10	457	Logical record length (word)
11	KD1SP	Number of physical record
12	Reserved	
13	"	
14	"	
15	"	
16	"	
17	"	
18	"	
19	"	
20	"	
21	"	
22	"	
23	"	
24	"	
25	"	
26	"	
27	"	
28	"	
29	"	
30	"	
31	"	
32	"	

Appendix 35 (Continued)

Word#	Content
33	Reserved
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"
46	"
47	"
48	"
49	"
50	"
51	"
52	"
53	"
54	"
55	"
56	"
57	"
58	"
59	"
60	"
61	"
62	"
63	"
64	"

Appendix 35 (Continued)

INCLOG - First physical record format

Word#	Content
1	I ; N .... File name
2	C ; L
3	O ; G
4	SP ; SP
5	Year .... Date
6	Month
7	Day
8	50 .... Physical record length (byte)
9	1 .... Number of physical record per logical record
10	25 .... Logical record length (word)
11	KINC .... Number of physical record
12	Reserved
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"
25	"

Appendix 35 (Continued)

ALMREC - First physical record format

Word#	Content	
1	A ; L	File name
2	M ; R	
3	E ; C	
4	SP ; SP	
5	Year	Date
6	Month	
7	Day	
8	28	Physical record length (byte)
9	1	Number of physical record per logical record
10	14	Logical record length (word)
11	NALREC	Number of physical record
12	Reserved	
13	"	
14	"	

Appendix 35 (Continued)

XMTTAB - First physical record format

Word#	Content
1	X ; M .... File name
2	T ; T
3	A ; B
4	SP ; SP
5	Year .... Date
6	Month
7	Day
8	102 .... Physical record length (byte)
9	1 .... Number of physical record per logical record
10	51 .... Logical record length (word)
11	NMSGTX .... Number of physical record
12	Reserved
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"
25	"
26	"
27	"
29	"
30	"
31	"
32	"

Appendix 35 (Continued)

Word#	Content
33	Reserved
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"
46	"
47	"
48	"
49	"
50	"
51	"

Appendix 35 (Continued)

RESDEV - First physical record format

Word#	Content
1	R ; E .... File name
2	S ; D
3	E ; V
4	SP ; SP
5	Year .... Date
6	Month
7	Day
8	52 .... Physical record length (byte)
9	1 .... Number of physical record per logical record
10	26 .... Logical record length (word)
11	- .... Number of physical record
12	Reserved
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"
25	"
26	"

Appendix 35 (Continued)

WEATHER - First physical record format

Word#	Content
1	W ; E .... File name
2	A ; T
3	H ; E
4	SP ; SP
5	Year .... Date
6	Month
7	Day
8	72 .... Physical record length (byte)
9	1 .... Number of physical record per logical record
10	36 .... Logical record length (word)
11	NWREC .... Number of physical record
12	Reserved
13	"
14	"
15	"
16	"
17	"
18	"
19	"
20	"
21	"
22	"
23	"
24	"
25	"
26	"
27	"
28	"
29	"
30	"
31	"
32	"

Appendix 35 (Continued)

Word#	Content
33	Reserved
34	"
35	"
36	"

Appendix 35 (Continued)

MSGFILE - File Logical Record Format

Word#	Content
1	Left byte : binary zero; Right byte : modem address
2	Left byte : SOH; Right byte : P
3	Left byte : 0; Right byte : @
4	Left byte : 0; Right byte : STX
5	2 message characters
6	" "
7	" "
8	" "
9	" "
10	" "
11	" "
12	" "
13	" "
14	" "
15	" "
16	CR; LF
17	2 message characters
18	" "
19	" "
20	" "
21	" "
22	" "
23	" "
24	" "
25	" "
26	" "
27	" "
28	ETX; NULL
29	Unused
30	"
31	"
32	"

Appendix 35 (Continued)

Word#	Content
33	Unused
34	"
35	"
36	"
37	"
38	"
39	"
40	"
41	"
42	"
43	"
44	"
45	"

```
C
      PROGRAM GEN
C
C
      EXTERNAL CLOCK
      EXTERNAL DINITIALIZE,DHEADING
C
C
C
C
C
C                                  OPEN OVERLAY FILE
C
      CALL OVOPN (1, 'GEN.OL', IER)
C
C                                  INITIALIZE TASKS AND VARIABLES
C
      CALL FOVLD(1,DINITIALIZE,0,IER)
      CALL INITIALIZE
      CALL FOVRL(DINITIALIZE,IER)
C
C                                  PRINT HEADING
C
      CALL FOVLD(1,DHEADING,0,IER)
      CALL HEADING
      CALL FOVRL(DHEADING,IER)
C
C                                  DEFINE USER CLOCK
C
      CALL DUCLK (10, CLOCK, IER)
C
C                                  GET OPERATOR COMMANDS
C
      CALL PRI (7)
      CALL TOPCOM
C
C                                  CLOSE OVERLAY FILE
C
      CALL CLOSE (1, IER)
C
C                                  END OF ROUTINE
C
      STOP
      END
```

```
C       SUBPROGRAM GENBLK
C       BLOCK DATA
C
C       THIS IS THE BLOCK DATA SUB-PROGRAM FOR 'WEM'.
C
C
C       COMMON STATEMENTS:
C
C       COMMON /ALARM/ NOBEEP, ALMBEEP
C         USED IN: ALM30S.ALM5M.AMODN.T17A.TWANSF
C
C       COMMON /ALM1/ APARM
C         USED IN: ALM30S.ALM5M.MXFER
C
C       COMMON /BLDE1/ FT140
C         USED IN: T14C.T140.TOPCOM
C
C       COMMON /COMM/ DETECT.MNETRC.MOKEY
C         USED IN: COMCON.DATAIN.MODM1.RAMCON.T18A
C
C       COMMON /CREST/ N.M
C         USED IN: RAMPS.RDISMAP
C
C       COMMON /CTRL1/ CONTRL.MAXCYC.MCTPR.MONCYC
C         USED IN: DIS2.GMAP.QUOCC.RAMCON.RCTIME.T12A.T13.UPD30S.WOOD30
C
C       COMMON /CTRL2/ CYCOVR.RAMCYC
C         USED IN: ALM30S.DIS2.QUOCC.RAMCON.RCTIME.T12A.UPD30S.WOOD30
C
C       COMMON /CTRL3/ CNTADM.CSTAT.LTSTAT
C         USED IN: ALM30S.RAMCON.T17A.T17B.T18A
C
C       COMMON /CTRL4/ CHEK1.CHEK2.LIGHTS
C         USED IN: ALM30S.MXFER.RAMCON
C
C       COMMON /CTRL5/ THRCP.CTRMOD
C         USED IN: RCTIME.T12A
C
C       COMMON /CTRL6/ COMDOP.CNTUM.MBRAMP
C         USED IN: COMCON.RAMCON.T18A
C
C       COMMON /CTRL7/ ERRCNT.DFREQ
C         USED IN: DATAIN. RAMCON
C
C       COMMON /CTRL9/ MCNT
C         USED IN: DATAIN.RAMCON
C
C       COMMON /D1S/ VOL1S.OCC1S.DVOL1S.UVOL1S.DOCC1S.UOCC1S.VSPEED.LVC1S.LTCOND
C         USED IN: DATAIN.MXFER
C
C       COMMON /D30/ DATA30
C         USED IN: ALM30S.DIS1.DIS2.DVS1/INCDET.INITIALIZE.MXFER.PFRCHK.QUOCC.RCTIME.
C           T140.UPD30S.WOOD30
C
C       COMMON /DIMDS/ XREFTB.XREF
C         USED IN: ALM30S.ALM5M.MODM1.MXFER.T140.T17B.UPD30S.WOOD30
C
C       COMMON /ERRP/ PR1ERR.PR2ERR.PR3ERR.PR4ERR.PR5ERR
C         USED IN: COMCON.MODM1
C
C       COMMON /ET/ DTIM1.OTIM1.DTIM2.UPTIM
C         USED IN: DATAIN.MXFER
C
C       COMMON /FXTWN/ WINDOW
C         USED IN: DISMAP.IOC2.SPD2
C
C       COMMON /GEOM/ DTUT.DTOT.NRAMPS.NSTNS.CAP
C         USED IN: ACCIDENT.ALM30S.ALM5M.CHOOR.DATAIN.DIS1.DIS2.DVS1.
C           FACC.FLASH.GMAP.INCDET.MXFER.PERCHK.QOCC.QSPR.QUOCC.
C           RAMCON.RAMPS.RCTIME.T140.UPD30S.WOOD30
C
C       COMMON /GRAPH/ MAPMOD.COLOR
C         USED IN: DISMAP.GMAP.RAMPS.SPD2.T19
C
C       COMMON /INCA/ INCTV.INCREC.INCFLAG.INCSTAT
C         USED IN: CLOW.DATSTR.DIS2.INCDET.PFRCHK.T16A.VTIM
C
C       COMMON /INCB/ OCCDNI.OCCUNI.INCNT.INCONT.INCEX
C         USED IN: INCDET.PERCHK.T17A.TWANSF
C
C       COMMON /INCC/ THRMAL
C         USED IN: INCDET.PERCHK
C
C       COMMON /INCD/ INCPHG.INCOT.INCVT.INCAT
```

```
C       USED IN: CLOU,DATSTR,INCDET,VTLU
C
C     COMMON /INCE/ CHCK,CMNI1,CMNJ,TEST3
C       USED IN: INCDET,PERCHK
C
C     COMMON /INCID/ ACC,ALOC,AFLG
C       USED IN: ACCIDENT,DETECTOR,FACC,FLASH,GMAP
C
C     COMMON /KLOCK/ TIMER
C       USED IN: CLOCK,COMCOM
C
C     COMMON /LASTMAP/ LMAP
C       USED IN: GMAP
C
C     COMMON /LIMITS/ MAXOCC,MAXSPD,IMIN,MAXVEH,STOLN,S1,S2
C       USED IN: DATAIN,DLCHEK,MODM1,NXFER,SLCHEK
C
C     COMMON /LOOPL/ LDIST
C       USED IN: DATAIN
C
C     COMMON /MD/ MODEMT,MODEM,MODEM1,NUMMDM
C       USED IN: AMDOM,COMCOM,GMAP,MODM1,T14C,T17A,T17B,T18A
C
C     COMMON /MDMD/ MODMD
C       USED IN: AMDOM
C
C     COMMON /MDTBL/MERROR,MDMDET,XTB
C       USED IN: MODM1
C
C     COMMON /NACC/ ECNT
C       USED IN: FLASH,LOOP
C
C     COMMON /NWCC/ [illegible]
C       USED IN: ALM3SS,ALMSM,AMDOM,CLOU,DATSTR,DIS1,DIS2,DYS1,HFADING,
C         INCDET,INITIALIZE,PERCHK,REGEN,T16A,T16B,T16C,T16D,
C         T18A,T18B,T18C,UPD3SS,VTLU,WOOD30
C
C     COMMON /OCCOM/ OCC,SMETER
C       USED IN: DISMAP,GMAP,GOCC,RDISMAP
C
C     COMMON /OPTS/ [illegible]
C       USED IN: DISMAP,GMAP,SPD1,SPD2
C
C     COMMON /OPTB/ IODL,IOCL,IOPL
C       USED IN: DISMAP,GMAP,IOCC
C
C     COMMON /PROT/ COMMSG,REPLY,PROTOCOL,MSGID,KTIME,PSPRT
C       USED IN: COMCOM,MODM1
C
C     COMMON/QENDATA/ QEW
C       USED IN: ACCIDENT,BOX2,BOX3,CHOOR,DISMAP,IOC2,IOCC,GOCC,GSPD,
C         REGEN,SPD1,SPD2
C
C     COMMON /QQ1/ QOVR,QIND
C       USED IN: CHOOR,QUOCC,RCTIME,T13
C
C     COMMON /QQ2/ [illegible]
C       USED IN: DIS2,QUOCC,RCTIME,T13,UPD3SS,WOOD30
C
C     COMMON /QQ3/ KSCNT,KSTEP,KTLMT,KTHR
C       USED IN: QUOCC,T13
C
C     COMMON /QQ4/ QOCCLB,QOCCUB
C       USED IN: CHOOR,QUOCC,T13
C
C     COMMON /QOC/ QOCC
C       USED IN: DIS2,QUOCC,RCTIME,T13
C
C     COMMON /RANGE/ UPRNGE,DNRNGE
C       USED IN: DLCHEK,SLCHEK,T18A
C
C     COMMON /RDET/ MPD,MDD,MED,MOD
C       USED IN: ALMSM,DIS1,DYS1,QUOCC,RCTIME
C
C     COMMON /RDFIL/ RDRCD,RSDVR
C       USED IN: CLOU,T1AC,VTLU
C
C     COMMON /RDOR/ MOTIM,INML,IMAB
C       USED IN: CHOOR,RAMPS
C
C     COMMON /SCREEN/ [illegible]
C       USED IN: BOX1,BOX2,BOX3,DETECTOR,DISMAP,FLASH,GMAP,IOC2,IOCC,
C         LOOP,GOCC,GSPD,RAMPS,RDISMAP,REGEN,SPD1,SPD2,TRANSF
C
      COMMON /SECTL/ DIST
```

```
C       USED IN: INCDET,PENCHM,UPD308,WOOD10
C
C     COMMON /BGN1/ INTBUF,RCVBUF,BGNADR
C       USED IN: ALM308,COMCOM,GMAP,INITIALIZE,T14A,T14B,T14C,T14D,T18B
C
C     COMMON /BGN2/ NWORDS,BUFSIZ,NWBGTX,NWBGRX,MTLBGN
C       USED IN: COMCOM,DIB2,GMAP,INITIALIZE,T14A,T14B,T14C,T14D,T18C
C
C     COMMON /BGN3/ BGNSTAT,ALMFLG,RCVTO,PAPLG,AMODE
C       USED IN: ALM308,COMCOM,GMAP,T14C,T14D,T17A,T17B
C
C     COMMON /BT18/ LATST,ERRARD,IDATA
C       USED IN: DATAIN,DLCHEK,SLCHEK
C
C     COMMON /TC/ IMIN10
C       USED IN: TRANSF
C
C     COMMON /TCNT/ COUNT,KCNT5M,IMINS,KCOUNT
C       USED IN: ALMSM,DIB1,DYB1,TRANSF,UPD308,WOOD30
C
C     COMMON /TSKBYN/ TSK0,TSK1,TSK2,TSK3
C       USED IN: ALMBGN,COMCOM,DATAIN,GMAP,T14D,TRANSF
C
C     COMMON /TTB/ II
C       USED IN: INCDET,PENCHM,UPD308,WOOD10
C
C     COMMON /WKDAY/ HMS,MDY
C       USED IN: ALM308,ALMSM,AMODM,CLDV,DATSTR,DIB1,DYB1,HEADING,INCDET,INITTALZIE,
C         PENCHM,RCTIME,T14D,T16A,T16B,T16C,T16D,T18B,T18C,
C         TRANSF,UPD308,WOOD10
C
C     COMMON /WORDS/ NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,KI
C       USED IN: [illegible],IOCC,INCF,
C         OOCC,SSPD,RAMPS,WDISMAP,SPD1,SPD2
C
C     COMMON /WPV/ WEATHER
C       USED IN: HEADING,T16A,T18B
C
C
C
C     DECLARATION STATEMENTS:
C
C     LOGICAL ACC(22)                       !/INCID/
C       TRUE IF ACCIDENT OCCURRED AT THIS STATION
C     LOGICAL AFLG                          !/INCID/
C       TRUE IF ACCIDENT PRESENT AT 1 OR MORE STATIONS
C     INTEGER ALMBEEP                       !/ALARM/
C       ?ALARM BEEP
C     INTEGER ALOC(3)                       !/INCID/
C       ACCIDENT LOCATIONS
C     INTEGER ALMFLG                        !/BGN3/
C       ?ALARM FLAG
C     INTEGER AMODE                         !/BGN3/
C       ?ALARM MODE
C     INTEGER APARM(6)                      !/ALM1/
C       ?ALARM PARAMETERS
C     INTEGER BUFSIZ                        !/BGN2/
C       BUFFER SIZE (IN WORDS) OF CMS MESSAGE
C     INTEGER CAP(22)                       !/GEOM/
C       CAPACITY OF EACH STATION
C     INTEGER CHEK(21)                      !/SNDC/
C       ?
C     INTEGER CHEK1(10),CHEK2(10)           !/CTRL4/
C       CHECKS FOR TRANSMISSION ERRORS WITH EACH RAMP CONTROLLER
C     INTEGER CHNI1(21),CHNJ(21)            !/INCF/
C       SAVE I1,J
C     INTEGER CNTADR(10)                    !/CTRL3/
C       CONTROLLER ADDRESS FOR EACH RAMP
C     INTEGER CNTON(10)                     !/CTRL6/
```

```
C        = 1 IF CONTROLLER ON FOR THIS RAMP
C
      INTEGER COLOR                         !/GRAPH/
C       COLOR
C
      INTEGER COMOUP(8)                     !/CTRL6/
C       ?COMMAND OPERATOR
C
      INTEGER COMMND(2)                     !/PRNT/
C       CONTROL MESSAGE TO ISI/SM 310 HEAD-END MODEM
C
      INTEGER CONTRL                        !/CTRL1/
C       CONTROL
C
      INTEGER COUNT                         !/TCNT/
C
      INTEGER CSTAT(10)                     !/CTRL3/
C       CONTROLLER STATUS (0=OK, 1=FAIL) OF EACH RAMP
C
      INTEGER CTRMOD                        !/CTRL5/
C       ?CONTROLLER MODE
C
      INTEGER CYCOVR(10)                    !/CTRL2/
C       CYCLE OVER-RIDE OF EACH RAMP
C
      INTEGER DATA30(710)                   !/D30/
C       RECORD IN FILE 'DATA30'
C
      INTEGER DETECT(545)                   !/COMM/
C       DETECTOR DATA
C
      INTEGER DFREQ                         !/CTRL7/
C       ?DATA FREQUENCY
C
      INTEGER DIST(20)                      !/SECTL/
C       DISTANCE (IN METRES)
C
      INTEGER DRANGE(67,3)                  !/RANGE/
C       DOWNSTREAM RANGE FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER DOCC1S(67)                    !/D1S/
C       DOWNSTREAM 1-SEC. OCCUPANCY VALUES FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER DTIM1(67),DTIM2(67)           !/ET/
C       AVERAGE ELAPSED TIMES FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER DTOT                          !/GEOM/
C       TOTAL NUMBER OF DOUBLE-LOOP DETECTORS
C
      INTEGER DVOL1S(67)                    !/D1S/
C       DOWNSTREAM 1-SEC. VOLUME VALUES FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER ERRAND                        !/ST1S/
C       ?
C
      INTEGER ERRCNT(10)                    !/CTRL7/
C       ERROR COUNT OF EACH RAMP
C
      INTEGER FT14D                         !/BLOK1/
C       FLAG TO SUBR 'FT14D'
C
      INTEGER GCHN                          !/SCREEN/
C       GRAPHICS CHANNEL
C
      INTEGER HMS(3)                        !/WKDAY/
C       HOUR, MINUTE, SECOND
C
      INTEGER IDATA                         !/ST1S/
C       ?
C
      INTEGER IMIN                          !/LIMITS/
C       COUNTER THAT RE-SETS EVERY 1 MINUTE
C
      INTEGER IMIN5                         !/TCNT/
C       COUNTER THAT RE-SETS EVERY 5 MINUTES
C
      INTEGER IMIN10                        !/TC/
C       COUNTER THAT RE-SETS EVERY 10 MINUTES
C
      LOGICAL IRMU(10)                      !/RQUM/
C       TRUE IF THIS RAMP IN QUEUE OVER-RIDE AND ABOVE UPPER BOUND
C
      LOGICAL IRML(10)                      !/RQUM/
C       TRUE IF THIS RAMP IN QUEUE OVER-RIDE AND BELOW LOWER BOUND
```

```
C       INTEGER INCONT
C         INCIDENT                         !/INCA/
C       INTEGER INCER
C         INCIDENT ?                       !/INCA/
C       INTEGER INCFLAG(21)
C         INCIDENT FLAG FOR EACH ?         !/INCA/
C       INTEGER INCNT(21)
C         INCIDENT COUNT FOR EACH ?        !/INCA/
C       INTEGER INCOT,INCST,INCVT
C         INCIDENT OCCUPANCY, SPEED, VOLUME TOTALS  !/INCD/
C       INTEGER INCPOS(21)
C         INCIDENT FOR EACH ?              !/INCD/
C       INTEGER INCREC(21)
C         INCIDENT RECORD FOR EACH ?       !/INCA/
C       INTEGER INCSTAT(21)
C         INCIDENT STATUS FOR EACH ?       !/INCA/
C       INTEGER INCTV(84)
C         INCIDENT ?                       !/INCA/
C       INTEGER IO,IS,IV                   !/WORDS/
C         INDICES IN 'DEM' RECORD FOR MAINLINE STATION OCCUPANCY, SPEED, AND VOLUME
C       INTEGER IOCL,IODL,IOPL             !/OPTA/
C         INDICES IN 'DEM' RECORD FOR OCCUPANCY IN CENTRE, DRIVING, AND PASSING LANES
C       INTEGER ISCL,ISDL,ISPL             !/OPTA/
C         INDICES IN 'DEM' RECORD FOR SPEED IN CENTRE, DRIVING, AND PASSING LANES
C       INTEGER K,KI                       !/WORDS/
C         CONTINUATION CHARACTERS FOR GRAPHICS COMMANDS
C       INTEGER KSM                        !/NRFC/
C         NUMBER OF RECORDS IN FILE 'DATASM'
C       INTEGER KCNT
C         ACCIDENT COUNT                   !/NACC/
C       INTEGER KCNTSM
C         15-MIN. COUNTER                  !/TCNT/
C       INTEGER KCNTO
C         ?                                !/BGNS/
C       INTEGER KCOUNT
C         ?                                !/TCNT/
C       INTEGER KDISP                      !/NRFC/
C         NUMBER OF RECORDS IN FILE 'DEMDTA'
C       INTEGER KINC                       !/NRFC/
C         NUMBER OF RECORDS IN FILE 'INCLOG'
C       INTEGER KL30S                      !/NRFC/
C         NUMBER OF RECORDS IN FILE 'DATA30'
C       INTEGER KQCNT(10)
C         QUEUE COUNT ?                          !/QDS/
C       INTEGER KSTEP(10)
C         STEP FOR EACH RAMP               !/QDS/
C       INTEGER KTMR
C         ?                                !/QDS/
C       INTEGER KTIME                      !/PROT/
C         10-MIN. COUNTER (RE-SET VIA 'PSPRT')
C       INTEGER KTLMT(10)
C         TIME LIMIT FOR EACH RAMP         !/QDS/
C       INTEGER LATST                      !/STIN/
C         LANE TEST: =1 IF NOT TO COMPUTE SPEED
C       INTEGER LDIST(67)                  !/LOOPL/
C         LOOP DISTANCES (IN 10*METRES) FOR EACH DOUBLE-LOOP DETECTOR
```

```
      INTEGER LIGHTS(10)                     !/CTRL4/
C       LIGHTS FOR EACH RAMP
C
      INTEGER LMAP                           !/LASTMAP/
C       LAST MAP
C
      INTEGER LTCOND(10)                     !/DIS/
C       LIGHT CONDITIONS FOR EACH RAMP
C
      INTEGER LTSTAT(10)                     !/CTRL3/
C       LIGHT STATUS FOR EACH RAMP
C
      INTEGER LVC18(67)                      !/DIS/
C       1-SEC. LANE VOLUME COUNTS FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER M,N                            !/CRFST/
C       INDICES IN 'SEN' RECORD OF FROM AND TO RAMP NUMBERS
C
      INTEGER MAPNUM                         !/GRAPH/
C       MAP NUMBER
C
      INTEGER MAXCYC                         !/CTRL1/
C       MAXIMUM NUMBER OF CYCLES
C
      INTEGER MAXOCC                         !/LIMITS/
C       MAXIMUM OCCUPANCY
C
      INTEGER MAXSPD                         !/LIMITS/
C       MAXIMUM SPEED
C
      INTEGER MAXVEH                         !/LIMITS/
C       MAXIMUM NUMBER OF VEHICLES
C
      INTEGER MCNT(10)                       !/CTRL9/
C       MAXIMUM COUNT FOR EACH RAMP
C
      INTEGER MDMDET(73)                     !/MDTAL/
C       DETECTOR LOCATION OF EACH MODEM
C
      INTEGER MDY(3)                         !/WKDAY/
C       MONTH, DAY, YEAR
C
      INTEGER METER(10)                      !/CTRL1/
C       = 1 IF METERING ON THIS RAMP
C
      INTEGER MODCYC(10)                     !/CTRL1/
C       MODEM CYCLE FOR EACH RAMP
C
      INTEGER MODEM(140)                     !/MD/
C       MODEM DATA
C
      INTEGER MODEMI(73)                     !/MD/
C       ?
C
      INTEGER MODEMT(299)                    !/MD/
C       MODEM TEST DATA
C
      INTEGER MODNO(73)                      !/MDNO/
C       NUMBER OF EACH MODEM
C
      INTEGER MSGID                          !/PROT/
C       MESSAGE IDENTIFIER
C
      INTEGER MTLSGN(2)                      !/SGN2/
C       3 FOR EACH SIGN
C
      INTEGER NALREC                         !/NRFC/
C       NUMBER OF RECORDS IN FILE 'ALMREC'
C
      INTEGER NBRAMP                         !/CTRL6/
C       = 1 IF NORTH-BOUND RAMP
C
      INTEGER [illegible]                    !/[illegible]/
C       NUMBER OF RAMP CONTROLLER POLYGONS ON INTERCHANGE MAP
C
      INTEGER NUD,NED                        !/NDFT/
C       NUMBER OF DEMAND, EXIT DETECTORS
C
      INTEGER NDETRC                         !/COMM/
C       NUMBER OF DETECT RECORDS
C
      INTEGER NOKEY                          !/COMM/
C       MESSAGE KEY BETWEEN 'COMCUM' & 'TIRA'
C
      INTEGER NERROR(73)                     !/MDTAL/
```

```
C        NUMBER OF ERRORS FOR EACH MODEM
C
      INTEGER NMSGRX,NMSGTX                 !/SGN2/
C        NUMBER OF CMS MESSAGES RECEIVED, TRANSMITTED
C
      INTEGER NUBEEP                        !/ALARM/
C        TWO BEEP
C
      LOGICAL [illegible](10)               !/[illegible]/
C        TRUE IF THIS RAMP IN QUEUE OVER-RIDE
C
      INTEGER NPD,NQD                       !/NDFT/
C        NUMBER OF PASSAGE, QUEUE DETECTORS
C
      INTEGER NRAMPS                        !/GEOM/
C        NUMBER OF RAMPS
C
      INTEGER NSTA                          !/WORDS/
C        NUMBER OF DETECTOR STATIONS FOR INTERCHANGE MAP
C
      INTEGER NSTNS                         !/GEOM/
C        NUMBER OF MAINLINE DETECTOR STATIONS
C
      INTEGER NUMMOD                        !/MO/
C        NUMBER OF MODEMS
C
      INTEGER NWORDS                        !/SGN2/
C        NUMBER OF WORDS IN CMS MESSAGE
C
      INTEGER NWREC                         !/WREC/
C        NUMBER OF RECORDS IN FILE 'WEATHER'
C
      LOGICAL OCC                           !/OCCRW/
C        TRUE IF COLOR BASED ON OCCUPANCY
C
      INTEGER OCC1S(46)                     !/D1S/
C        1-SEC. OCCUPANCY VALUES FOR EACH SINGLE-LOOP DETECTOR
C
      INTEGER OCCDSI(21),OCCUSI(21)         !/INCB/
C        DOWNSTREAM, UPSTREAM OCCUPANCY FOR EACH ?
C
      INTEGER P4FLG                         !/SGN3/
C        PROTOCOL 4 FLAG
C
      INTEGER PFILE                         !/SCREEN/
C        CHANNEL FOR POLYGON FILES
C
      INTEGER PN1ERR,PN2ERR,PN3ERR,PN4ERR,PN5ERR      !/ERRP/
C        ERROR COUNT FOR PROTOCOLS 1,2,3,4,5
C
      INTEGER PROTOCOL                      !/PROT/
C        PROTOCOL NUMBER
C
      INTEGER PSPRT                         !/PROT/
C        NUMBER OF SECONDS IN 10 MIN. (RE-SETS 'KTIME')
C
      INTEGER [illegible]                   !/QEWDATA/
C        RECORD IN FILE 'QEWDYN'
C
      INTEGER QIND(10)                      !/QD1/
C        = 1 IF QUEUE OVER-RIDE FOR THIS RAMP
C
      INTEGER QLNTH(10)                     !/QD2/
C        QUEUE LENGTH FOR EACH RAMP
C
      INTEGER QOCC(13)                      !/QOC/
C        QUEUE OCCUPANCY
C
      INTEGER QOCCLB(10),QOCCUB(10)         !/QD4/
C        QUEUE OCCUPANCY LOWER, UPPER BOUND FOR EACH RAMP
C
      INTEGER QOVR(10)                      !/QD1/
C        QUEUE OVER-RIDE FOR EACH RAMP
C
      INTEGER QWAIT(10)                     !/QD2/
C        QUEUE WAIT FOR EACH RAMP
C
      INTEGER RAMCYC(10)                    !/CTRL2/
C        CYCLE FOR EACH RAMP
C
      INTEGER RCVBUF(64)                    !/SGN1/
C        RECEIVE BUFFER FROM CMS
C
      INTEGER RDRCD                         !/RDFIL/
C        NUMBER OF RECORDS IN FILE 'RESDEV'
C
```

```
      INTEGER REPLY(2)                    !/PROT/
C       REPLY FROM IBI/3M 310 HEAD-END MODFM
C
      LOGICAL RMETER(10)                  !/OCCRM/
C       TRUE IF METERING ON THIS RAMP
C
      INTEGER RO,RV                       !/WORDS/
C       INDICES IN 'SEW' RECORD OF RAMP OCCUPANCY AND VOLUME VALUES
C
      INTEGER NOEB                        !/SCREEN/
C       CHANNEL FOR 'SEWDYN' FILE
C
      INTEGER NSDVR(21)                   !/NDFIL/
C       ?
C
      INTEGER CHTEMP                      !/SCREEN/
C       CHANNEL FOR 'TEMPFILE' FILE
C
      INTEGER S1,S2                       !/LIMITS/
C       ?
C
      INTEGER SGNADR(2)                   !/SGN1/
C       ADDRESS OF EACH SIGN
C
      INTEGER SGNSTAT(2)                  !/SGN%/
C       STATUS OF EACH SIGN
C
      INTEGER STDLN                       !/LIMITS/
C       STANDARD LENGTH (IN 10*M.) OF VEHICLE
C
      INTEGER STIM1(46)                   !/ET/
C       1-SEC. TIMER FOR EACH SINGLE-LOOP DETECTOR
C
      INTEGER STOT                        !/SEDM/
C       TOTAL NUMBER OF SINGLE-LOUP DETECTORS
C
      INTEGER TEST3                       !/INCE/
C       3RD TEST STATISTIC FOR INCIDENT DETECTION & CONFIRMATION
C
      INTEGER THRESP(4)                   !/CTRLS/
C       ?
C
      INTEGER THRMOL                      !/INCC/
C       ?
C
      INTEGER TIMER                       !/KLOCK/
C       MESSAGE KEY FROM 'CLOCK' TO 'COMCON'
C
      INTEGER TSK0                        !/TSKSYN/
C       MESSAGE KEY FROM 'DATAIN' TO 'ALMSGN'
C
      INTEGER TSK1                        !/TSKSYN/
C       MESSAGE KEY FROM 'DATAIN' TO 'TRANSF'
C
      INTEGER TSK2                        !/TSKSYN/
C       MESSAGE KEY FROM 'TRANSF' TO 'SMAP'
C
      INTEGER TSK3                        !/TSKSYN/
C       MESSAGE KEY FROM 'COMCON' TO 'TIOD'
C
      INTEGER TT(20)                      !/TTS/
C       ?
C
      INTEGER TXT(4)                      !/WORDS/
C       TEXT FOR SPEED, VOLUME, OCCUPANCY
C
      INTEGER WOCC1S(67)                  !/D1S/
C       1-SEC. OCCUPANCY DATA FOR DOUBLE-LOOP DETECTORS
C
      INTEGER UPRNGE(113,2)               !/RANGE/
C       UPSTREAM RANGE FOR EACH DETECTOR
C
      INTEGER UPTIM(67)                   !/ET/
C       UPSTREAM TIME FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER UVOL1S(67)                  !/D1S/
C       1-SEC. VOLUME DATA FOR EACH DOUBLE-LOOP DETECTOR
C
      INTEGER VOL1S(46)                   !/D1S/
C       1-SEC. VOLUME DATA FOR EACH SINGLE-LOOP DETECTOR
C
      INTEGER VSPEED(67)                  !/D1S/
C       1-SEC. VEHICLE SPEED DATA FOR EACH DOUBLE-LOOP DFTFCTOR
C
```

```
      INTEGER WEATHR(54)              !/WPV/
C        RECORD IN FILE 'WEATHER'
C
      LOGICAL WINSTON                 !/EXTRA/
C        TRUE IF EXTRA LANE FOR WINSTON CHURCHILL BLVD. DISPLAYED
C
      INTEGER XMTBUF(45)              !/BGM1/
C        TRANSMIT BUFFER TO CMB
C
      INTEGER XREF(113)               !/DINDX/
C        CROSS-REFERENCE FOR EACH DETECTOR
C
      INTEGER XREFID(100)             !/DINDX/
C        CROSS-REFERENCE TABLE FOR EACH ?
C
      INTEGER XTB(100)                !/MOTHL/
C        ? TABLE
C
C
C
C     DATA STATEMENTS:
C
C
      DATA [illegible] /0/
      DATA ALMFLG /0/
      DATA AMODE /0/
      DATA APARM(1) /14.713.75.7125.1.250/
      DATA BUFSIZ /45/
      DATA CAP(1) /22*6000/
      DATA CHEK1(1) /10*0/, CHEK2(1) /10*0/
      DATA CHTADR(1) /60.64.8.12.20.26.32.40.40.48/
      DATA [illegible] /10*1/
      DATA COLOR /1/
      DATA CONTRL /0/
      DATA COUNT /1/
      DATA CTRMOD /0/
      DATA CYCOVR(1) /10*0/
      DATA DATA30(1) /71*0/
      DATA DFREQ /30/
      DATA [illegible]
     *   / 700, 737, 970, 649, 990,1005, 837, 742, 729, 682,
     *     920, 732, 797, 825, 749, 743, 784, 845, 819, 886/
      DATA DRANGE(1,1) /201*0/
      DATA DOCC15(1) /67*0/
      DATA DTIM1(1) /67*0/, DTIM2(1) /67*0/
      DATA DTOT /67/
      DATA DVOL15(1) /67*0/
      DATA [illegible]
      DATA ERRCNT(1) /10*0/
      DATA FT100 /0/
      DATA ICMB /14/
      DATA IDATA /0/
      DATA IMIN /3/
      DATA IMINS /0/
      DATA IMIN15 /0/
      DATA [illegible]
      DATA INCSI /0/
      DATA INCFLAG(1) /21*0/
      DATA INCNT(1) /21*0/
      DATA INCUT /2/, INCST /2/, INCVT /9/
      DATA INCPRS(1) /21*0/
      DATA INCREC(1) /21*0/
      DATA INCSTAT(1) /21*0/
      DATA [illegible]
     *    /0:5:46:17, 0:5:46:17, 0:5:46:17, 0:5:46:17
     *    ,0:5:46:17, 0:5:46:17, 0:5:46:17, 0:5:46:17
     *    ,0:5:46:17, 0:5:46:17, 0:5:46:17, 0:5:46:17
     *    ,0:5:46:17, 0:5:46:17, 0:5:46:17, 0:5:46:17
     *    ,0:5:46:17, 0:5:46:17, 0:5:46:17, 0:5:46:17
     *    ,0:5:46:17/
      DATA K /'&&'/, K1 /'& '/
      DATA [illegible]
      DATA KCNISM /0/
      DATA KCNTB /0/
      DATA KCOUNT /0/
      DATA KDISP /1/
      DATA KINC /1/
      DATA KLSOS /1/
      DATA KOCNT(1) /10*0/
      DATA [illegible]
      DATA KTMR /60/
      DATA KTIME /0/
      DATA KTLMT(1) /10*0/
      DATA LATST /0/
      DATA LUTB(1)
     * /70.61.61.55.55.55.61.61.61.60
     * .57.57.55.55.55.61.61.61.60.60
     * .60.61.61.61.61.61.61.62.60.61
     * .61.61.59.60.61.62.61.58.61.61
     * .61.61.62.61.62.59.61.61.61.59
     * .61.61.55.55.56.62.61.62.61.61
     * .60.56.60.59.58.60.59/
      DATA [illegible] /10*0/
      DATA LMAP /99/
      DATA LTCOMB(1) /10*0/
      DATA LVSTAT(1) /10*0/
      DATA LVSIB(1) /67*0/
      DATA MAPNUM /1/
      DATA MAXCYC /5/
      DATA MAXOCC /25/
      DATA [illegible]
      DATA MAXVEH /6/
      DATA MCNT(1) /0.0.1.2.3.4.5.6.6.7/
      DATA MDMODE(1)
     * /10*0.16.3*0.55.106.166.226.4*0.285.3*0
     * .355.[illegible].596.626.2*0.686.726.0.787
     * .8[illegible].3*0.916.996.1056.5*0.1136.1176.1256.1296
     * .13[illegible].0.1376.1436.6*0.1496.1556.1617.3*0.1636
     * [illegible]
      DATA MDTEN(1) /10*0/
      DATA MUDCYC(1) /10*0/
      DATA MUDEM(1) /100*0/
      DATA MUDEM[illegible] /73*0/
      DATA MUDFMT(1) /99*0/
      DATA MUDNU(1)
     * /8*0.4*1.4*2.3.4.3*0.4*5.4*6.7.8.9.10.4*11.12
     * [illegible]
     * .23.0.4*24.25/
      DATA MOGID /-1/
      DATA MTLBGN(1) /2*0/
      DATA NALMES /1/
      DATA NBRAMP /0/
      DATA NDD /10/, NED /11/
      DATA NDETRC /0/
      DATA [illegible]
      DATA NEWMOD(1) /73*0/
      DATA NMSGRI /0/, NMSGTX /0/
      DATA NODEEP /0/
      DATA NPD /10/, NRD /13/
      DATA NRAMPS /10/
      DATA NSTNS /2/
      DATA NUMMOD /3/
      DATA [illegible]
      DATA NWREC /1/
      DATA OCC /.TRUE./
      DATA OCC1D(1) /66*0/
      DATA OCCUBI(1) /21*0/, OCCUBI(1) /21*0/
      DATA PGFLE /0/
      DATA PFILE /7/
      DATA PR1ERR /0/, PR2ERR/0/, PR3ERR /0/, PR4ERR/0/, PR5ERR /0/
      DATA [illegible]
      DATA PSPWT /600/
      DATA QINU(1) /10*0/
      DATA QLMTM(1) /10*0/
      DATA QOCC(1) /13*0/
      DATA QOCCLB(1) /10*450/, QOCCUB(1)/ 10*600/
      DATA QUVR(1) /10*0/
      DATA QWAIT(1) /10*0/
      DATA [illegible] /10*0/
      DATA RORCD /1/
      DATA RUER /15/
      DATA RUTEMP /9/
      DATA S1 /6/, S2 /0/
      DATA SGMADR(1) /1.2/
      DATA SGMSTA(1) /2*0/
      DATA STDLM /16/
      DATA [illegible]
      DATA STOT /96/
      DATA TMMCP(1) /4*0/
      DATA TMMMDL /675/
      DATA TIMER /0/
      DATA TSK0 /0/
      DATA TSK1 /0/
      DATA TSK2 /0/
      DATA [illegible]
      DATA TT(1) /20*0/
      DATA TXT(1) /' S','PU',' V','UL',' O','CC'/
      DATA UUCC1B(1) /67*0/
      DATA UPRNGE(1.1) /226*0/
      DATA [illegible]
      DATA UVOLIB(1) /67*0/
      DATA VOLIB(1) /[illegible]/
      DATA VSPEED(1) /67*0/
      DATA WEATHER(1) /6*0,'MI',9*0,'PI',9*0,'VI',9*0/
      DATA SHEF(1)
     * / 3,13,23,39, 4,14,24,33,25, 5,15,34,45,[illegible], 6,16
     * ,37,38,26, 7,17,39,40, 8,18,41, 9,19,42,28,29
     * [illegible]
     * ,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
     * ,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44
     * ,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60
     * ,61,62,63,64,65,66,67,10,11,12, 7, 8, 9, 4, 5, 6, 1, 2, 3/
      DATA SHEFTO(1)
     * /513,516, [illegible]
     * [illegible]
     * [illegible]
     * ,528,530,[illegible],522,524,526,508,510,512,502,504,506
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible],131,133,137,139,141,143,145,107/
      DATA STO(1)
     * / [illegible],127, 60,128, 61
     * [illegible]
     * [illegible]
     * [illegible],136, 69,137, 70
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible],158, 91
     * [illegible],161, 94
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * [illegible]
     * ,114, 47,115, 48,116, 49/
C
      END
      OVERLAY UACCIDENT
      SUBROUTINE ACCIDENT
C
C
C FUNCTION : LOOP THROUGH THE STATIONS TO DETERMINE WHERE AN ACCIDENT
C            HAS OCCURED. THE LOGICAL VARIABLE ACC HAS BEEN SET TO .FALSE.
C            IF INCIDENT =1 ACC WILL BE REDEFINED AS .TRUE. INDICATING
C            AN ACCIDENT HAS BEEN DETECTED. A FLAG "AFLG" WILL BE SET TO TRUE
C
C
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/MEMDATA/GEN
      COMMON/INCIO/ACC,ALOC,AFLG
      INTEGER GEN(5,17),ALOC(3)
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      LOGICAL ACC(17),AFLG
      AFLG=.FALSE.
C
C     SET ALL STATIONS TO BE ACCIDENT FREE
C
      DO 30 I=1,NSTNS
   30 ACC(I)=.FALSE.
C
C     LOOP TO FIND ACCIDENT  I2=INITIAL WORD FOR INCIDENT
C
      I2=17
      DO 40 I=1,NSTNS
      I2=I2-5
      IF(GEN(I2) .EQ. 1) ACC(I)=.TRUE.
      IF(GEN(I3) .EQ. 1) AFLG=.TRUE.
   40 CONTINUE
      RETURN
      END
      OVERLAY DALMSON
      SUBROUTINE ALMS08
C
      COMMON/DSN1/DNTUMP(45),RCVMUP(45),DSNADR(2)
      COMMON/DSN3/DSNSTAT(2),ALMFLG,KCNTD,PAFLG,AMODE
      COMMON/ALM1/APARM(6)
      COMMON/CTRL3/CHTABM(10),CSTAT(10),LTSTAT(10)
      [illegible]
      COMMON/ESM/DATA30(716)
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/DIAG3/SREFTB(100),SREF(113)
      COMMON/MREC/ALS08,NMREC,KINC,KSM,DALREC,KDISP
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/ALARM/NBEEP,ALMBEEP
      COMMON/HRDAY/HMS(3),MDY(3)
      INTEGER ALMREC(16)
      INTEGER SSPD(30),DNOCC(67),UPOCC(67),DVOL(10)
      INTEGER DNTUMP,RCVMUP,DSNADR
      INTEGER DSNSTAT,ALMFLG,KCNTD,PAFLG,AMODE
      INTEGER APARM
      INTEGER CHTABM,CSTAT,LTSTAT
      INTEGER CHEK1,CHEK2,LIGHTS
      [illegible]
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER SREFTB,SREF
      INTEGER ALS08,NMREC,KINC,KSM,DALREC,KDISP
      INTEGER CYCOVR,RAMCYC
      INTEGER NBEEP,ALMBEEP
      INTEGER HMS,MDY
      INTEGER I1,I2,J,ID,IOCC,IER,IBEEP,LNO
      [illegible]
C
      EQUIVALENCE (DNOCC(1),DATA30(273)),(UPOCC(1),DATA30(362))
      EQUIVALENCE (SSPD(1),DATA30(675)),(DVOL(1),DATA30(591))
C
C
C
C OPEN FILE
      CALL OPEN(6,"ALMREC",3,IER,80)
C
C
C PROGRAM ID AND CHECKING PERIOD
C
      ALMREC(13)=1 ;PROG. ID
      ALMREC(14)=30 ;CHECK PERIOD IN SEC
C
      ALMREC(1)=HMS(1)
      ALMREC(2)=HMS(2)
      ALMREC(3)=HMS(3) ;TIME IT FAILS
C
C
C CHECK DOUBLE-LOOP DETECTORS FOR HALFN IF STN SPEED GE. 50KM/H(14M/S)
C
      IBEEP=0
      DO 100 I=1,NSTNS
      [illegible]
      IF(SSPD(J).LT.APARM(1))GOTO 100 ;STN SPEED<50KM/H,NO CHECK
      I1=3*I-2 ;INDEX TO LOCATE CORR. MAINLINE DET OCC
      I2=I1+2
      IF(J.LE.6)I2=I2+1 ;STN 6 HAS 4 LANES
      IF(J.LT.6)I1=I1+1
      LNO=0 ;INIT LANE NO
      DO 90 K=I1,I2
      [illegible]
      DO 92 II=1,2 ;1 FOR DOWN.,2 FOR UPSTREAM
      ID=STOT+K+(II-1)*DTOT ;INDEX TO LOCATE CORR DET ADDR
      IOCC=DNOCC(K) ;CORR DOWNSTR. OCC
      IF(II.EQ.2)IOCC=UPOCC(K) ;CORR UPSTR OCC
      IF(SREFTB(ID).GE.100)GOTO 92 ;ERROR ALR. NOTED
      IF(IOCC.LE.APARM(2))GOTO 92 ;OCC LE 95%,OK
      ALMREC(5)=SREFTB(ID) ;DET ADD THAT FAILS
      [illegible] ;DET FAIL FLAG SET
      ALMREC(4)=1 ;ID FOR DET FAIL
      ALMREC(6)=J ;STN NO
      ALMREC(7)=LNO ;LANE NO
      ALMREC(8)=II ;UP OR DOWNSTREAM
      ALMREC(9)=1 ;OCC WRONG
      ALMREC(10)=0 ;NOT USED
      ALMREC(11)=1 ;TYPE OF DATA(OCC)
      [illegible]
      CALL WRITR(6,DALREC,ALMREC,1,IER)
      DALREC=DALREC+1
      IBEEP=1 ;ALARM FLAG SET
 92   CONTINUE
 90     CONTINUE
100     CONTINUE
C
C
C  CHECK CONTROLLER FOR MALFN
C
        DO 700 I=1,NRAMPS
        CSTAT(I)=0
        IF(CHEK1(I).EQ.CHEK2(I))GOTO 700
        DO 710 BIT=0,15
        ID=3 ;DEV ID FOR C170
        IF(I.GE.3.AND.I.LE.7)ID=7 ;DEV ID FOR CD710
        IF(ID.EQ.7.AND.(BIT.EQ.4.OR.BIT.GE.14))GOTO 710 ;BIT04-13 CHECKED(710)
        DIBIT=0
        COBIT=0
        IF(ITEST(CHEK1(I),BIT))DIBIT=1
        IF(ITEST(CHEK2(I),BIT))COBIT=1
        IF(DIBIT.EQ.COBIT)GOTO 710 ;ERROR NOT IN THIS BIT POSITION
        ALMREC(10)=BIT ;REC BIT POS
        ALMREC(4)=ID ;DEVICE ID
        ALMREC(5)=CNTADR(I) ;CONTROLLER ADDR
        ALMREC(6)=I ;CONT LOCATION
        ALMREC(7)=0 ;NOT USED
        ALMREC(8)=0
        ALMREC(9)=0
        ALMREC(11)=0
        ALMREC(12)=0
        CALL WRITR(8,MALREC,ALMREC,1,IER)
        MALREC=MALREC+1
        IBEEP=1 ;ALARM FLAG SET
710     CONTINUE
        CSTAT(I)=1 ;CONTROLLER FAIL FLAG SET
700     CONTINUE
C
C  CHECK LIGHT STATUS
C
        DO 800 I=1,NRAMPS
        IF(LTSTAT(I).GE.3)GOTO 800 ;ERROR ALR NOTED
        GR=0
        AM=0
        RD=0
        IF(ITEST(LIGHTS(I),10))GR=2
        IF(ITEST(LIGHTS(I),9))AM=3
        IF(ITEST(LIGHTS(I),8))RD=4
        SUM=GR+AM+RD
        IF(RAMCYC(I).NE.0)GOTO 900 ;RAMP IS METERED WITH NON GREEN
        IF(SUM.EQ.2)GOTO 820 ;NO ERROR
        ALMREC(11)=1 ;P.GREEN CYCLE
        GOTO 810 [illegible]
900     IF(SUM.EQ.6.OR.SUM.EQ.9)GOTO 820 ;NO ERROR
        IF(DVOL(I).LE.0)GOTO 820 ;NO VEH THRU SIGNAL LAST 30 SECS
        ALMREC(11)=2 ;NORMAL CYCLE
810     ALMREC(4)=4 ;DEV ID
        ALMREC(5)=CNTADR(I)+1 ;DEVICE ADDR
        ALMREC(6)=I ;DEV LOCATION
        ALMREC(7)=0
        ALMREC(8)=0 ;NOT USED
        ALMREC(9)=SUM ;ERR CODE
        ALMREC(10)=0 ;NOT USED
        ALMREC(12)=0
        LTSTAT(I)=LTSTAT(I)+1
        IF(LTSTAT(I).LT.3)GOTO 800
        IBEEP=1 ;ALARM FLAG SET
        WRITE(10,830)I,SUM ;3 CONSECUTIVE ERROR,NOTIFY OPERATOR
830     FORMAT([illegible] LIGHT FAIL [illegible] 'CODE' [illegible])
840     CALL WRITR(8,MALREC,ALMREC,1,IER)
        MALREC=MALREC+1
        GOTO 800
820     LTSTAT(I)=0
800     CONTINUE
C
C  LOOK FOR SIGN MALFN: RECORD TO 'ALMREC' IF FAIL FLAG SET
C
        IF(ALMFLG.EQ.0)GOTO 400
        ALMFLG=0 ;MALFN ACK'ED.RESET FLAG
        ALMREC(4)=6 ;DEV ID
        ALMREC(5)=SGNADR(1) ;DEV ADDR
        ALMREC(6)=1 ;DEV LOCATION
        DO 410 I=7,12
410     ALMREC(I)=0 ;NOT USED
        CALL WRITR(8,MALREC,ALMREC,1,IER)
        MALREC=MALREC+1
C
C
400     IF(IBEEP.EQ.1)ALMBEEP=1 ;NOTIFY OPERATOR
```

```
      COMPILER NOSTACK
      OVERLAY OALMSM
      SUBROUTINE ALMSM
C
      COMMON/ALM1/APARM(6)
      COMMON/GEOM/STOT,OTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/DIMNS/SREFTB(160),SREF(113)
      [illegible]
      COMMON/ALARM/NODEEP,ALMDEEP
      COMMON/NKDAY/NMS(3),NDT(3)
      COMMON/TCNT/COUNT,KCNT5M,IMINS,KCOUNT
      COMMON/NDET/NPD,NDD,NED,NOD
C
      INTEGER ALMREC(14),DATA5M(448)
      INTEGER APARM
      INTEGER [illegible]
      INTEGER EXVOL(11)
      INTEGER SREFTB,SREF
      INTEGER AL300,NNREC,SINC,KSM,NALREC,KDISP
      INTEGER NODEEP,ALMDEEP
      INTEGER NMS,NDT
      INTEGER COUNT,KCNT5M,IMINS,KCOUNT
      INTEGER NPD,NDD,NED,NOD
      INTEGER [illegible],IDET1,IDET2,IDET3,IPOS
      INTEGER ICODE,J,I1,I2,LNG,ID
C
      DATA EXVOL(1) /11*0/
C
C
C
C OPEN FILE
C
      CALL OPEN(8,"ALMREC",3,IER,2*14)
C
C
      ALMREC(13)=2 ;PROG ID
      ALMREC(1)=NMS(1)
      ALMREC(2)=NMS(2)
      ALMREC(3)=NMS(3)
      [illegible]
      ALMREC(10)=0 ;NOT USED
      ALMREC(14)=300 ;CHECKING PERIOD IN SECS
      IBEEP=0
C
C OPEN FILE TO GET 5-MIN DATA
C
      NREC5M=7
      [illegible]
      CALL READR(8,N,DATA5M,7,IER)
      CALL CLOSE(8,IER)
C
C ACC 15-MIN EXIT VOL
C
      IF(KCNT5M.GE.3)KCNT5M=0 ;RESET TIME COUNTS EVERY 15 MINS
      DO 80 I=1,NED
      [illegible]
      IDX=I+NPD+NDD+3 ;INDEX FOR EXIT VOL
      IF(DATA5M(IDX).LT.0)EXVOL(I)=-1
80    IF(EXVOL(I).GE.0)EXVOL(I)=EXVOL(I)+DATA5M(IDX)
      KCNT5M=KCNT5M+1 ;5-MIN COUNTS
C
C CHECK SINGLE LOOP DETECTORS
C
      IDET1=NPD+NDD
      IDET2=IDET1+NED
      IDET3=IDET2+NOD
      DO 100 I=1,STOT
      IDX=I+STOT+3 ;OCC INDEX
      IF(SREFTB(I).GE.1000.OR.DATA5M(IDX).LT.0)GOTO 100 ;ERR ALR NOTED
      IF(I.LE.NPD)GOTO 110 ;PASSAGE DET
      IF(I.LE.IDET1)GOTO 120 ;DEMAND DET
      IF(I.LE.IDET2)GOTO 130 ;EXIT DET
      IF(I.LE.IDET3)GOTO 140 ;QUEUE DET
      ALMREC(7)=5 ;MAINLINE SINGLE-LOOP DET
      IPOS=I-IDET3
      GOTO 112
140   IPOS=I-IDET2 ;QUEUE DET
      ALMREC(7)=4
      GOTO 112
110   IPOS=I
      ALMREC(7)=1
112   ICODE=1 ;OCC ERR CODE
      IF(DATA5M(IDX).LT.APARM(1).OR.DATA5M(IDX).GT.APARM(4))GOTO 199
      IDX=I+3 ;VOL INDEX
      [illegible].LE.APARM(4))GOTO 190
      ICODE=2 ;VOL ERR CODE
      GOTO 190
140   ALMREC(7)=2
      IPOS=I-NPD ;DET LOC
      IF(DATASM(IDX).LE.APARM(3))GOTO 180
      ICODE=1 ;OCC ERR CODE
      GOTO 190
180   ALMREC(7)=1
      ICODE=1
      IPOS=I-[illegible]
      IF(DATASM(IDX).GT.APARM(4))GOTO 190
      IF(ECNT[illegible].LT.3)GOTO 190 ;NOT 15 MINS YET,NO VOL CHECK
      IPOS=I+[illegible] ;VOL INDEX
      IF(EIVOL(IPOS).GE.APAR4(5))GOTO 190
      ICODE=2
170   [illegible] ;DEV ADDR
      [illegible]+1000
      ALMREC(6)=IPOS+1000 ;DET LOC
      ALMREC(8)=0 ;NOT USED
      ALMREC(9)=ICODE ;ERR CODE
      ALMREC(11)=ICODE ;TYPE OF DATA
      ALMREC(12)=DATASM(IDX) ;VALUE
      CALL WRITE(6,NALREC,ALMREC,1,IER)
      NALREC=NALREC+1
      IGEEP=1 ;ALARM FLAG SET
190   CONTINUE
C
C CHECK MAINLINE DETECTORS
C
      DO 600 I=1,NSTNS
      J=NSTNS-I+1 ;STN NO
      [illegible]
      I2=I1+2
      IF(J.LT.4)I2=I2+1 ;STN 4 HAS 4 LANES
      IF(J.LT.[illegible])I1=I1+1
      LNO=0 ;INIT LANE NO
      DO 500 K=I1,I2
      LNO=LNO+1
      DO 400 L=1,2 ;1 FOR DOWN.,2 FOR UPSTREAM
      [illegible] ;INDEX TO GET CORR MAINLINE DET
      [illegible] ;INDEX TO LOC CORR DET NO
      IF([illegible].GE.1000.OR.DATASM(IDX).LT.0)GOTO 400 ;ERR ALR NOTED
      ICODE=1 ;OCC ERR CODE
      IF(DATASM(IDX).LT.APARM(3).OR.DATASM(IDX).GT.APARM(4))GOTO 410
      IDX=IDX+[illegible] ;MAINLINE VOL INDEX
      IF(DATASM(IDX).GE.APARM(5).AND.DATASM(IDX).LE.APARM(6))GOTO 400
      ICODE=2 ;VOL ERR CODE
410   [illegible] ;DEV ADDR
      [illegible]+1000 ;SET ERR FLAG SET
      ALMREC(6)=J ;STN NO
      ALMREC(7)=LNO ;LANE NO
      ALMREC(8)=L ;UP OR DOWNSTR
      ALMREC(9)=ICODE ;ERR CODE
      ALMREC(11)=ICODE ;TYPE OF DATA
      ALMREC(12)=DATASM(IDX) ;VALUE
      CALL WRITE(6,NALREC,ALMREC,1,IER)
      NALREC=NALREC+1
      IGEEP=1
400   CONTINUE
500   CONTINUE
600   CONTINUE
      IF(IGEEP.EQ.1)ALMBEEP=1
      CALL CLOSE(6,IER)
C
      RETURN
      END
```

```
C
C     TASK ALMSGN
C
C     A ONE-MINUTE PERIODIC CORRIDOR TRAFFIC SURVEILLANCE
C       AND CONTROL TASK, INCLUDING HIGHWAYS 2 AND 5 PCS,
C       DATA LINK TO MISSISSAUGA TRAFFIC CONTROL COMPUTER
C       SYSTEM, AND AUTOMATIC CMS CONTROL.
C
C
C     SPECIFICATION STATEMENTS:
C
C     COMMON STATEMENTS:
C
      COMMON /TSKSTA/ TSK0,TSK1,TSK2,TSK3
C
C     DATA-TYPE STATEMENTS:
C
C                      INPUT VARIABLES
C
      INTEGER TSK0,TSK1,TSK2,TSK3
C
C                      INTERNAL VARIABLES
C
      INTEGER ALSGCT
      INTEGER MSG
C
C
      ALSGCT = 0
C
C     WAIT FOR UNT FROM SUBR 'DATAIN'
C
 100  CONTINUE
      CALL REC (TSK3, MSG)
C
C     INCREMENT EXECUTION COUNTER
C
      ALSGCT = ALSGCT + 1
      GO TO 1000
C
C     INVOKE 1 MIN. ROUTINES
C
 1000 CONTINUE
C
C     INVOKE PCS POLLING ROUTINE
C
      CALL PCSPOL
C
C     INVOKE MISSISSAUGA COMMUNICATION ROUTINE
C
      CALL MISSCM
      GO TO 2000
C
C     INVOKE 5 MIN. ROUTINES
C
 2000 CONTINUE
      IF (.NOT. (MOD(ALSGCT, 5) .EQ. 0) ) GO TO 3000
C
C     INVOKE CMS CONTROL ROUTINE
C
      CALL CMSCTR
C
C     INVOKE CORRIDOR CONTROL ROUTINE
C
      CALL CORCTR
C
C     INVOKE 10 MIN. ROUTINES
C
 3000 CONTINUE
      IF (.NOT. (MOD(ALSGCT, 10) .EQ. 0) ) GO TO 9000
C
C     INVOKE ARTERIAL SIGNAL ROUTINE
C
      CALL ARTSIG
C
```

[illegible]

```
      OVERLAY [illegible]
      SUBROUTINE AMODM
      DIMENSION MODEMM(148),ALMREC(16)
      COMMON/MD/MODEMT(299),MODEM(148),MODFM1(73),NUMMDM
      COMMON/MREC/ALJOB,NMREC,AINC,ASM,NALREC,ADISP
      COMMON/MDNO/MODNO(73)
      COMMON/ALARM/MODEEP,ALMEEP
      COMMON/HRDAY/HMS(3),MDY(3)
      INTEGER MODEMM,ALMREC
      INTEGER MODEMT,MODEM,MODFM1,NUMMDM
      INTEGER ALJOB,NMREC,AINC,ASM,NALREC,ADISP
      INTEGER MODNO
      INTEGER MODEEP,ALMEEP
      INTEGER HMS,MDY
      INTEGER IM1,IO1,IO2,IER
      EQUIVALENCE (MODEMM(1),MODEMT(152))
C
C WRITE OUT DATE & TIME INFORMATION
C
      WRITE(10,90)MDY,HMS
90    FORMAT(//I5,'/',I2,'/',I2,5X,I2,':',I2,':',I2//'>'Z)
C OPEN FILE
      CALL OPEN(0,"ALMREC",3,IER,28)
C
C PROGRAM ID AND CHECKING PERIOD
C
      ALMREC(13)=3 ;PROGRAM ID
      ALMREC(14)=600 ;CHECKING PERIOD IN SEC
C
      ALMREC(1)=HMS(1)
      ALMREC(2)=HMS(2)
      ALMREC(3)=HMS(3) ; TIME IT FAILS
      ALMREC(4)=3 ;DEVICE ID FOR MODEMS
C
C CHECK MODEMS FOR MALFUNCTION
C
      DO 100 I=1,NUMMDM
      IF(I.LE.8) GOTO 100 ;NOT USED
      IF([illegible])GOTO 100 ;NOT USED
      IF([illegible])GOTO 100 ;NOT USED
      IF([illegible])GOTO 100 ;NOT USED
      IF(MODEMM(I).GE.10)GOTO 100; ERROR ALREADY DETECTED
      IF(MODEMT(I+33).LT.0.OR.MODEMM(I).GT.0)GOTO 110
      MODEM(I)=0
      GOTO 100
110   MODEM(I)=MODEM(I)+1
      IF(I.EQ.37)TYPE "HIGH MODEM MALFUNCTION"
      IF(MODEM(I).EQ.1)ALMEEP=1
      IM1=I-1
      IO1=IM1/8
      IO2=IO1/8
      ALMREC(5)=(IO2*10+MOD(IO1,8))*10+MOD(IM1,8) ;MODEM ADDR IN OCTAL NO
      ALMREC(6)=MODNO(I) ;MODEM NO (L[illegible]TION)
      ALMREC(7)=0
      ALMREC(8)=0
      ALMREC(9)=1 ;ERROR CODE
      IF(MODEMM(I).GT.0)ALMREC(9)=2
      IF(MODEMM(I).GT.0.AND.MODEMT(I+33).LT.0)ALMREC(9)=3
      ALMREC(10)=0
      ALMREC(11)=0
      ALMREC(12)=MODEMM(I)
      CALL WRITR(0,NALREC,ALMREC,1,IER)
      NALREC=NALREC+1
100   CONTINUE
      CALL CLOSE(0,IER)
      RETURN
      END
      SUBROUTINE ARTSIG
C
C     A TEN-MINUTE PERIODIC ARTERIAL SIGNALIZATION CONTROL PROGRAM.
C
C
      RETURN
      END
      OVERLAY UBOX1
      SUBROUTINE BOX1
C
C
C FUNCTIONS - READS NUMBER OF MAINLINE STATIONS AND TOTAL NUMBER OF RAMP
C                 AND EXIT CONTROLLERS FOR THIS MAP
C             - SETS COLOUR TO DRAW BOXES
C             - READS THE ACTUAL INITIAL ELEMENTS FOR MAINLINE STATION SPEED
C                 AND RAMP VOLUME
C             - DISPLAYS BOXES IN GREEN
C
C      NLIGHTS = THE NUMBER OF CYCLE LENGTHS TO BE DISPLAYED FOR THIS MAP
C         CLEN = THE INITIAL WORD POSITION IN GENDYN FOR CYCLE LENGTH
C        NPOLY = TOTAL NUMBER OF   POLYGONS TO BE DISPLAYED(NSTA+NCON)
C        NSTA  = NUMBER OF MAINLINE POLYGONS TO CONTAIN TEXTUAL DATA
C        NCON  = NUMBER OF RAMP & EXIT CONTROLLERS CONTAINING TEXTUAL DATA
C        IS = INITIAL WORD IN GENDYN FOR SPEED AT THE STATION IN DURATION
C        IV =                                VOLUME
C        IO =                               OCCUPANCY
C        RV =                             RAMP VOLUMES
C        RO =                            RAMP OCCUPANCIES
C
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      COMMON/SCREEN/GCHN,PFILE,RHTEMP
      INTEGER NSTA,NCON,IS,IV,IO
      INTEGER RV,RO,TXT(6),GCHN,PFILE,RHTEMP,IDATA(39)
C
C     READ NUMBER OF RAMP CONTROLLERS FOR THIS DISPLAY
C
      READ(PFILE,500) NCON
500   FORMAT(I2)
      NPOLY=NSTA+NCON
C
C     SET COLOUR FOR TEXT BOXES
C
      WRITE(GCHN,600)
600   FORMAT(" ","ICOL C2 C0")
C
C     READ ACTUAL INITIAL SUBSCRIPTS FOR MAINLINE STATION SPEED
C     AND RAMP VOLUME
C
      READ(PFILE,700)IS,RV
700   FORMAT(2(I3,1X))
      IV=IS+1
      IO=IV+1
      RO=RV+1
C
C     DISPLAY MAINLINE RAMP & EXIT VECTOR BOXES
C
      DO 100 I=1,NPOLY
15    READ(PFILE,10) (IDATA(J),J=1,39)
      WRITE(GCHN,35) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+66A
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 15
10    FORMAT(39A2)
35    FORMAT(" ",39A2)
100   CONTINUE
      RETURN
      END
      OVERLAY OBOX2
      SUBROUTINE BOX2
C
C
C    FUNCTION : TO DISPLAY THE MAINLINE AND RAMP STATION DATA
C      FOR ALL THE DISPLAYS EXCEPT STATION 15A ON MISS RD.
C
C     NSTA  = NUMBER OF MAINLINE POLYGONS TO CONTAIN TEXTUAL DATA
C     NCON  = NUMBER OF RAMP & EXIT CONTROLLERS CONTAINING TEXTUAL DATA
C     IS = INITIAL WORD IN QEWDYN FOR SPEED AT THE STATION IN QUESTION
C     IV =                                 VOLUME
C     IO =                               OCCUPANCY
C     RV =                             RAMP VOLUMES
C     RO =                           RAMP OCCUPANCIES
C
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,RQEW,RWTEMP
      INTEGER NSTA,NCON,IS,IV,IO
      INTEGER QEW(512),IDATA(94),RV,RO,TXT(6),GCHN,PFILE,RQEW,RWTEMP
55    FORMAT(6A2)
C
C     DISPLAY THE TEXTUAL DATA FOR THE MAINLINE STATIONS
C
C     SPEED
C
      DO 200 I=1,NSTA
      READ(PFILE,55) (IDATA(J),J=1,6)
      WRITE(GCHN,150)(IDATA(J),J=1,6),QEW(IS),TXT(1),TXT(2)
      IS=IS+5
C     VOLUME
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(GCHN,150)(IDATA(J),J=1,6),QEW(IV),TXT(3),TXT(4)
      IV=IV+5
C     OCCUPANCY
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(GCHN,150)(IDATA(J),J=1,6),QEW(IO),TXT(5),TXT(6)
      IO=IO+5
150   FORMAT(" ",6A2,"1STH /",I3,2A2,"/")
200   CONTINUE
C
C   DISPLAY RAMP & EXIT TEXTUAL DATA
C              -SINCE MISS RD SCREWS THINGS UP REDUCE NCON BY 1
C
      IF(MAPNUM .EQ. 5) NCON=NCON-1
      DO 300 I=1,NCON
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(GCHN,150)(IDATA(J),J=1,6),QEW(RV),TXT(3),TXT(4)
      RV=RV+2
C
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(GCHN,150)(IDATA(J),J=1,6),QEW(RO),TXT(5),TXT(6)
      RO=RO+2
300   CONTINUE
      RETURN
      END
```

```
C
      OVERLAY UROX3
      SUBROUTINE BOX3
C    FUNCTIONS DISPLAY CYCLE TIME FOR ON RAMPS AS WELL AS
C     STATION DATA FOR STA 15A ON MISS RD.
C
C    NLIGHTS = THE NUMBER OF CYCLE LENGTHS TO BE DISPLAYED FOR THIS MAP
C      CLEN = THE INITIAL WORD POSITION IN GENDYN FOR CYCLE LENGTH
C     NPOLY = TOTAL NUMBER OF   POLYGONS TO BE DISPLAYED(NSTA+NCON)
C     NSTA   = NUMBER OF MAINLINE POLYGONS TO CONTAIN TEXTUAL DATA
C     NCON   = NUMBER OF RAMP & EXIT CONTROLLERS CONTAINING TEXTUAL DATA
C     IS = INITIAL WORD IN GENDYN FOR SPEED AT THE STATION IN QUESTION
C     IV =                          VOLUME
C     IO =                         OCCUPANCY
C     RV =                       RAMP VOLUMES
C     RO =                      RAMP OCCUPANCIES
C
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      COMMON/GENDATA/GEN
      COMMON/SCREEN/SCHN,PFILE,RGEN,RWTEMP
      INTEGER NSTA,NCON,IS,IV,IO
      INTEGER GEN(512),IDATA(30),RV,RO,TXT(6),SCHN,PFILE,RGEN,RWTEMP
      INTEGER NLIGHTS,CLEN
      IF(NWHSP.EQ. 3) GOTO 1000    ;IF MISS RD. HAS STA 15A
C
C     DISPLAY CYCLE LENGTH
C
1100  READ(PFILE,800)NLIGHTS,CLEN     ;GET # OF CLENGTHS TO DISPLAY & INIT POSI
800   FORMAT(2I2)
      DO 850 I=1,NLIGHTS
      READ(PFILE,55)(IDATA(J),J=1,6)     ;READ VECTOR POSITION
      I100=GEN(CLEN)/100
      IP=GEN(CLEN)-(I100*100)
      I10=IP/10
      I1=IP-(I10*10)
      WRITE(SCHN,900) (IDATA(J),J=1,6),I100,I10,I1
      CLEN=CLEN+5
850   CONTINUE
900   FORMAT(" ",6A2,"18TH/",2I1,".",I1,"SEC/")
      RETURN
C
C  FOR STA 15A ON MISS RD
C
1000  IV=167
      IO=168
C
C     READ & DISPLAY MAINLINE STATION DATA
C
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(SCHN,150)(IDATA(J),J=1,6),GEN(IV),TXT(3),TXT(4)
C
C     READ CO-ORD FOR BOXES & DISPLAY TEXTUAL DATA
C
      READ(PFILE,55)(IDATA(J),J=1,6)
      WRITE(SCHN,150)(IDATA(J),J=1,6),GEN(IO),TXT(5),TXT(6)
      GOTO 1100
55    FORMAT(6A2)
150   FORMAT(" ",6A2,"STN/",I3,2A2,"/")
      END
      OVERLAY QCHOUR
      SUBROUTINE CHSOM
C
C     FUNCTION: TO DETERMINE IF THE RAMPS ARE IN A Q OVERRIDE CONDITION, IF SO
C              THEN TO WHAT DEGREE? THIS ROUTINE PASSES THESE CONDITIONS
C              THROUGH THE LOGICAL VARIABLES DESCRIBED BELOW.
C     CALLER: TASK QMAP
C
C     ARGUMENTS :
C              NOTIN(I) LOGICAL VARIABLE WHICH WILL BE PASSED AS FALSE IF THE
C                       RAMP INQUESTION IS NOT IN QUEUE O/R CONDITION.
C
C              INBL(10) LOGICAL VARIABLE TO PASS FALSE IF IN QUEUE O/R CONDITION
C                       BUT BELOW UPPER BOUND THRESHOLD VALUE.
C
C              INAB(10) LOGICAL VARIABLE TO PASS FALSE IF IN QUEUE O/R CONDITION
C                       BUT ABOVE UPPER BOUND THRESHOLD VALUE.
C
C        I INDICATES THE RAMPS FOR THE INTERCHANGE
C
C               1,2  CAWTHRA NB & SB
C               3,4  HWY 10 NB & SB
C               5,6  MISSISSAUGA NB & SB
C                 7  SOUTH DOWN NB
C               8,9  WINSTON CHURCHILL NB & SB
C                10  FORD DR
C
      COMMON/QCOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/QQ1/QOVR(10),QIND(10)
      COMMON/QQ4/QOCCLB(10),QOCCUB(10)
      COMMON/QEMDATA/QEM
      COMMON/NSOR/NOTIN,INBL,INAB
C
      INTEGER QOVR,QIND
      INTEGER QOCCLB,QOCCUB
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER QEM(512)
C
      LOGICAL NOTIN(10),INBL(10),INAB(10)
C
C
C
      DO 50 I=1,NRAMPS                             ;INIT TO TRUE
      NOTIN(I)=.TRUE.
      INAB(I)=.TRUE.
      INBL(I)=.TRUE.
50    CONTINUE
      J=10                                      ;FIRST WORD
      DO 150 I=1,NRAMPS
      IOCC=QOCCUB(I)*10/73                      ;CONVERT PULSES TO %
C
C     TEST FOR QUEUE O/R CONDITIONS
C
      IF(QIND(I) .NE. 1) NOTIN(I)=.FALSE.
      IF(QIND(I) .NE. 1) GOTO 75                   ;NOT IN Q O/R SO SKIP TESTS
      IF(QEM(J) .LT. IOCC) GOTO 100                 ;IN Q O/R BUT BELOW VALUE
      INBL(I)=.FALSE.
      IF(QEM(J) .GT. IOCC) GOTO 100                 ;IN Q O/R BUT ABOVE VALUE
      INAB(I)=.FALSE.
      GOTO 100
C
C     RAMP NOT IN Q O/R CONDITIONS SO SET NOTIN AS BEING FALSE
C
75    INBL(I)=.FALSE.
      INAB(I)=.FALSE.
100   J=J+2
150   CONTINUE
C
      RETURN
      END
      OVERLAY OCLOU
      SUBROUTINE CLOU
C
      COMMON/MISC/KL300,NNREC,KINC,KSM,NALREC,KDISP
      COMMON/MRDAT/HMS(3),MDY(3)
      COMMON/INCA/INCTV(66),INCREC(21),INCFLAG(21),INCSTAT(21)
      COMMON/INCO/INCPAG(21),INCOT,INCVT,INCST
      COMMON/[illegible]
C
      INTEGER KL300,HMS,MDY,INCTV,INCREC,INCFLAG,INCSTAT
      INTEGER INCOT,INCVT,INCST,NORCD,NSDVR
      INTEGER LNOCC(66),LNVOL(66),LNSPD(66),STNOCC(22),STNVOL(22),STNSPD(22)
      INTEGER DL,CL,PL,DNDL,DNCL,DNPL
      INTEGER MIN,SIGMA
      INTEGER NESDEV(26),DATAX(320)
      INTEGER STN,I,TREC,TOCC,TVOL,TSPD
      INTEGER DNOCC,DNVOL,DNSPD
C
C OPEN DISK FILE "DATAX" TO OBTAIN OCC., VOL., SPD. DATA AT TIME OF INCIDENT
C
      CALL OPEN (0,'DATAX',2,IER,128)
      DO 20 STN=1,21
        IF(STN.EQ.3)GOTO 20;                        STN. 3 HAS NO DNSTRM STN
        IF(INCSTAT(STN).EQ.INCPAG(STN))GOTO 20;     NO NEW INCIDENT HAS OCCURRED : GO TO NEXT STATION
          I=1
C
C SET TREC(STARTING RECORD TO BE READ IN DATAX) AND READ CORRESPONDING LOGICAL RECORD OF "DATAX"
C
 41       TREC=(I-1)*5
          CALL READR(0,TREC,DATAX,5,IER)
C
C BREAK DATAX ARRAY DOWN INTO SIX SMALLER ARRAYS (CONTAINING: OCC., VOL., SPD., DATA FOR LANES AND STATIONS
C
          DO 21 I=1,66
            LNOCC(I)=DATAX(I)
            LNVOL(I)=DATAX(66+I)
            LNSPD(I)=DATAX(132+I)
 21         CONTINUE
C
          DO 22 J=1,22
            STNOCC(J)=DATAX(198+J)
            STNVOL(J)=DATAX(220+J)
            STNSPD(J)=DATAX(242+J)
 22         CONTINUE
C
C CALCULATE ELEMENTS OF ARRAY CONTAINING LANES OF STATION (OF INCIDENT)
C
          PL=STN*3-2
          CL=STN*3-1
          DL=STN*3
C
C SET INDICES TO LOCATION OF STATION DOWNSTREAM FROM THE INCIDENT
C
          INCRMT=3
          IF(STN.EQ.3)INCRMT=6;                     STN #3 IS ON 403 THEREFORE STN #4 IS DNSTRM FROM STN # 2
C
          DNPL=PL+INCRMT
          DNCL=CL+INCRMT
          DNDL=DL+INCRMT
C
C OCCUPANCY CLOU
C SELECT LANE WITH MIN. DOWNSTREAM OCC.
C
          [illegible]
          IF(LNOCC(DNDL).LT.LNOCC(DNCL))GOTO 30     YES - CENTER LANE
            IF(LNOCC(DNCL).LT.LNOCC(DNPL))GOTO 32;  YES - DRIVING LANE
 30           IF(LNOCC(DNDL).LT.LNOCC(DNPL))GOTO 31;  NO  - PASSING LANE
                NESDEV(TOCC)=3;
                MIN=DNPL
                GO TO 33
 31           NESDEV(TOCC)=1                         DRIVING LANE
              MIN=DNDL;
              GO TO 33
 32         NESDEV(TOCC)=2;                          CENTER LANE
            MIN=DNCL
C
C CALCULATE SIGMA VALUE FOR LANE - TEST AGAINST LANE INCIDENT OCC. THRESHOLD VALUE
C
 33       IF(STNOCC(STN+1).LE.0)GOTO 50;            CHECK TO PREVENT DIVISION BY ZERO
            SIGMA=((STNOCC(STN+1)-LNOCC(MIN))*10)/(STNOCC(STN+1))
            IF(SIGMA.GE.INCOT)GOTO 40;              YES - LANE OF INCIDENT
C
C IF SIGMA TOO SMALL, SELECT LANE WITH MAX. UPSTREAM OCC.
C
```

```
34        IF(LNOCC(DL).GT.LNOCC(CL))GOTO 35
            IF(LNOCC(CL).GT.LNOCC(PL))GOTO 37;                  CENTER LANE
35            IF(LNOCC(DL).GT.LNOCC(PL))GOTO 36;                 DRIVING LANE
                RESDEV(TOCC)=3;                                    PASSING LANE
                GO TO 40
36            RESDEV(TOCC)=1;                                      DRIVING LANE
              GO TO 40
37          RESDEV(TOCC)=2;                                        CENTER LANE
C
C VOLUME CLOU
C SELECT LANE WITH MINIMUM VOLUME AT THE DOWNSTREAM STATION
C
40        TVOL=T+18
          IF(LNVOL(DNDL).LT.LNVOL(DNCL))GOTO 41
            IF(LNVOL(DNCL).LT.LNVOL(DNPL))GOTO 43;              YES - CENTER LANE
41            IF(LNVOL(DNDL).LT.LNVOL(DNPL))GOTO 42;             YES - DRIVING LANE
                RESDEV(TVOL)=3;                                    NO  - PASSING LANE
                MIN=DNPL
                GO TO 44
42            RESDEV(TVOL)=1
              MIN=DNDL;                                            DRIVING LANE
              GO TO 44
43          RESDEV(TVOL)=2;                                        CENTER LANE
            MIN=DNCL
C
C CAL. SIGMA - TEST AGAINST LANE INC. VOL. THRESHOLD
C
44        IF(STNVOL(STN+1).LE.0)GOTO 48;                         TEST FOR DIVISION BY ZERO
            SIGMA=((STNVOL(STN+1)-LNVOL(MIN))*10)/(STNVOL(STN+1))
            IF(SIGMA.GE.INCVT)GOTO 50;                           YES - LANE OF INCIDENT
C
C IF SIGMA TOO SMALL , SELECT LANE WITH MINIMUM UPSTREAM VOL.
C
48        IF(LNVOL(DL).LT.LNVOL(CL))GOTO 45
            IF(LNVOL(CL).LT.LNVOL(PL))GOTO 47;                  CENTER LANE
45            IF(LNVOL(DL).LT.LNVOL(PL))GOTO 46;                 DRIVING LANE
                RESDEV(TVOL)=3;                                    PASSING LANE
                GO TO 50
46            RESDEV(TVOL)=1;                                      DRIVING LANE
              GO TO 50
47          RESDEV(TVOL)=2;                                        CENTER LANE
C
C SPEED CLOW
C SELECT LANE WITH MINIMUM UPSTREAM SPEED
C
50        TSPD=T+22
          IF(LNSPD(DL).LT.LNSPD(CL))GOTO 51
            IF(LNSPD(CL).LT.LNSPD(PL))GOTO 53;                  YES - CENTER LANE
51            IF(LNSPD(DL).LT.LNSPD(PL))GOTO 52;                 YES - DRIVING LANE
                RESDEV(TSPD)=3;                                    NO  - PASSING LANE
                MIN=PL
                GO TO 54
52            RESDEV(TSPD)=1
              MIN=DL;                                              DRIVING LANE
              GO TO 54
53          RESDEV(TSPD)=2;                                        CENTER LANE
            MIN=CL
C
C CAL. SIGMA - TEST AGAINST LANE INC. SPD THRESHOLD
C
54        IF(STNSPD(STN).LE.0)GOTO 58
            SIGMA=((STNSPD(STN)-LNSPD(MIN))*10)/(STNSPD(STN))
            IF(SIGMA.GE.INCST)GOTO 60
C
C IF SIGMA TOO SMALL SELECT LANE WITH MAX. DNSTRM SPEED
C
58        IF(LNSPD(DNDL).GT.LNSPD(DNCL))GOTO 55
            IF(LNSPD(DNCL).GT.LNSPD(DNPL))GOTO 57;              CENTER LANE
55            IF(LNSPD(DNDL).GT.LNSPD(DNPL))GOTO 56;             DRIVING LANE
                RESDEV(TSPD)=3;                                    PASSING LANE
                GO TO 60
56            RESDEV(TSPD)=1;                                      DRIVING LANE
              GO TO 60
57          RESDEV(TSPD)=2;                                        CENTER LANE
C
C REPEAT PROCEDURES USING PREVIOUS TIME SLICE 'T-1'
C INCREMENT 'T' , DECREMENT 'RFC10' ,CHECK 'T' TO TEST IF THREE TIME SLICES HAVE BEEN TAKEN YET
C
60        T=T+1
          IF(T.LE.3)GOTO 61
C
C WRITE "RESDEV" ARRAY TO 'RESDEV' DISK FILE TO PASS VOTE DATA TO VTLU ROUTINE
C
          CALL OPEN(3,'RESDEV',2,IER,52)
          CALL WRITR(3,WDRCD,RESDEV,1,IER)
          CALL CLOSE(3,IER)
          NSDVN(STN)=RUNCD
          RUNCD=RUNCD+1
C
   40     CONTINUE
        CALL CLOSE(0,IER)
        CALL DFILW('DATAX',IER);                       "DATAX" IS A TEMPORARY FILE USED TO PASS DATA
C                                                      BETWEEN ROUTINES; AFTER USE IT IS DELETED
        RETURN
        END
```

```
      SUBROUTINE CMSCTW
C
C     A FIVE-MINUTE PERIODIC CMS SIGN CONTROL PROGRAM,
C       BASED ON THE SURVEILLANCE DATA FROM QEW (& 403)
C       AND HIGHWAYS 2 & 5 PCS.
C
      RETURN
      END
```

```
C
C       TASK COMCOM
C
C       A ONE-SECOND PERIODIC WAKEUP TO MOVE A/B
C          COMMUNICATION AND TRAFFIC DATA PROCESSING TASK.
C
C       LEGEND:
C          PROTOCOL #1 = RECEIVE DETECTOR DATA
C          PROTOCOL #2 = TRANSMIT RAMP CONTROL COMMANDS
C          PROTOCOL #3 = RECEIVE MODEM RESPONSE DATA
C          PROTOCOL #4 = TRANSMIT MESSAGE TO CMS
C          PROTOCOL #5 = RECEIVE MESSAGE ECHO-BACK FROM CMS
C
        COMMON/BLOCK/TIMER
        COMMON/[illegible]/TSK0,TSK1,TSK2
        COMMON/COMM/DETECT(505),NDETRC,NOKEY
        COMMON/PROT/COMMSG(2),REPLY(2),PROTOCOL,MSGID,KTIME,PRPNT
        COMMON/ERRP/PR1ERR,PR2ERR,PR3ERR,PR4ERR,PR5ERR
        COMMON/MD/MODEMT(299),MODEM(140),MODEM1(73),NUMMDM
        COMMON/CIMLG/COMDOP(8),CNTON(10),NBRAMP
        COMMON/SGN1/INTBUF(45),RCVBUF(45),SGNADD(2)
        COMMON/SGN2/NNORDS,BUFB(2),NNGSTS,NNSGRS,MTLBSN(2)
        COMMON/[illegible]
        COMMON/CNTFLG/NZERO2
C
        INTEGER MDMTMP(140)
        INTEGER TIMER
        INTEGER TSK0,TSK1,TSK2
        INTEGER DETECT, NDETRC, NOKEY
        INTEGER COMMSG,REPLY,PROTOCOL,MSGID,KTIME,PRPNT
        INTEGER [illegible]
        INTEGER MODEMT,MODEM,MODEM1,NUMMDM
        INTEGER COMDOP,CNTON,NBRAMP
        INTEGER INTBUF,RCVBUF,SGNADD
        INTEGER NNORDS,BUFB(2),NNGSTS,NNSGRS,MTLBSN
        INTEGER SGNSTAT,ALMFLG,KCNTO,PRFLG,AMODE
        INTEGER NZERO2
C
        EQUIVALENCE (MDMTMP(1),MODEMT(4))
C
C
C
50      CONTINUE
        CALL REC (TIMER, MSGOTH)
        IF(PROTOCOL.LT.1) PROTOCOL=1
        GO TO (100,200,300,400,500),PROTOCOL
C
100     COMMSG(1)=1 ;PROTOCOL=1
        COMMSG(2)=505
        CALL BWRIT(COMMSG(1),2,IST) ;XMIT CONTROL MESSAGE
        CALL BREAD(REPLY(1),2,IST) ;RECEIVE REPLY FROM 80C8P
        TYPE "COMMSG =", COMMSG, " REPLY =", REPLY
        IF (REPLY(2).NE.COMMSG(2))PR1ERR=PR1ERR+1
        IF (REPLY(2).NE.COMMSG(2))GO TO 100;ERROR-TRY AGAIN
        [illegible] ;GET DETECTOR DATA FROM 80C8P
        TYPE "DETECT(505) =", DETECT(505)
        IF (MSGID.EQ.32767)MSGID=-1 ;REINITIALIZE TO AVOID INTEGER OVERFLOW
        MSGID=MSGID+1 ;INC MESSAGE SEQUENCE NO
        PROTOCOL=0 ;PROTOCOL #1 IS COMPLETED
        IF (MSGID.LT.0)GO TO 900 ;IGNORE FIRST DETECT BUFFER
C
C       CHECK IF DETECTOR DATA TO BE STORED
C
        IF (.NOT. (NDETRC .GT. 0) )  GO TO 190
C
C          WRITE DETECTOR DATA TO FILE
C
           CALL WRITR (6, NDETRC, DETECT(1), 1, IER)
           NDETRC = NDETRC + 1
C
C          CHECK IF 60 SEC. ELAPSED
C
           IF (.NOT. (NDETRC .GT. 60) )  GO TO 190
C
C             RETURN TO DUNK. 'T10A'
C
              CALL XMT (NOKEY, 1, $150)
              GO TO 190
C
C             ERROR IN XMT CALL
C
150           TYPE "ERROR IN TASK 'COMCOM': NOKEY =", NOKEY
              GO TO 190
```

```
C              END IF
C
C
C           END IF
 190    CONTINUE
        CALL DATAIN
        CALL MUDMI
        CALL NAMCOM
C
        PROTOCOL=2
 200    COMMSG(1)=2 ;PROT #2
        COMMSG(2)=8
        CALL BWRIT(COMMSG(1),2,IST)
        CALL BREAD(REPLY(1),2,IST) ;RECEIVE REPLY FROM BOCSP
C       TYPE "COMMSG =", COMMSG, " REPLY =", REPLY
C       IF (REPLY(2).NE.COMMSG(2))P2ERR=P2ERR+1
C       IF (REPLY(2).NE.COMMSG(2))GO TO 200
        CALL BWRIT([illegible]) ;[illegible] COMMAND TO BOCSP
        PROTOCOL=0 ;PROTOCOL #2 COMPLETED
        IF (P4FLG.EQ.0)GO TO 360
        IF (P4FLG.GT.1)GO TO 420
C
        PROTOCOL=4 ;PROT #4
 400    COMMSG(1)=4
        COMMSG(2)=45
        CALL BWRIT(COMMSG(1),2,IST)
        CALL BREAD(REPLY(1),2,IST)
C       TYPE "COMMSG =", COMMSG, " REPLY =", REPLY
        IF (REPLY(2).NE.-2)GO TO 410
        AMODE=1
        PROTOCOL=0
        GO TO 570
 410    CONTINUE
C       IF (REPLY(2).NE.COMMSG(2))P4ERR=P4ERR+1
C       IF (REPLY(2).NE.COMMSG(2))GO TO 400
        CALL BWRIT(INTBUF(1),45,IST)
        PROTOCOL=0
 420    P4FLG=P4FLG+1
        IF (P4FLG.LT.3)GO TO 360
C
        PROTOCOL=5
 500    COMMSG(1)=5
        COMMSG(2)=45
        CALL BWRIT(COMMSG(1),2,IST)
        CALL BREAD(REPLY(1),2,IST)
C       TYPE "COMMSG =", COMMSG, " REPLY =", REPLY
        IF(REPLY(2).NE.-2)GO TO 510
        AMODE=1
        PROTOCOL=0
        GO TO 570
 510    CONTINUE
C       IF (REPLY(2).NE.COMMSG(2))P5ERR=P5ERR+1
C       IF (REPLY(2).NE.COMMSG(2))GO TO 500
        CALL BREAD(RCVBUF(1),45,IST)
        NMSG=NMSG+1
        PROTOCOL=0
        P4FLG=0
C
 570    CONTINUE
        CALL INT(TBKO,1,0540)
        NZERO2=0
        GO TO 360
C
 580    CONTINUE
        IF (NZERO2.EQ.0)
     *   TYPE "ERROR IN TASK 'COMCOM' FOR SUBR 'TP140'", ",
     *   "TBKO = ", TBKO
        NZERO2=1
        GO TO 360
C
 360    IF (STIME.LT.P3PRT)GO TO 900
        STIME=0
C
        PROTOCOL=3
 300    COMMSG(1)=3 ;PROT #3
        COMMSG(2)=100
        CALL BWRIT(COMMSG(1),2,IST) ;INIT PROT #3 TO BOCSP
        CALL BREAD(REPLY(1),2,IST) ;RECEIVE REPLY FROM BOCSP
C       TYPE "COMMSG =", COMMSG, "REPLY =", REPLY
        IF (REPLY(2).EQ.-1)GO TO 400 ;ABORT THIS PROT
C       IF (REPLY(2).NE.COMMSG(2))P3ERR=P3ERR+1
C       IF (REPLY(2).NE.COMMSG(2))GO TO 300
 800    CALL BREAD(MODINP(1),144,IST) ;GET MODEM STATUS
        PROTOCOL=0 ;PROT #3 COMPLETED
        GO TO 900
C
```

```
[illegible]   CONTINUE
              GO TO 50
              END
```

```
      SUBROUTINE LOBCTO
C
C     A FIVE-MINUTE PERIODIC CORRIDOR CONTROL ALGORITHM
C       CALCULATION AND PERFORMANCE EVALUATION PROGRAM,
C       BASED ON SURVEILLANCE DATA FROM QEW, PCS, AND
C       MISSISSAUGA TRAFFIC CONTROL COMPUTER SYSTEM.
C
      RETURN
      END
      SUBROUTINE DATAIN
C
C
      COMMON/TDADYN/IDN0,TDN1,TDN2,TDN3
      COMMON/SCOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/C15/VOL15(46),OCC15(46),DVOL15(67),UVOL15(67),DOCC15(67),
     *   UOCC15(67),VSPEED(67),LVC15(67),LTCONT(10)
      COMMON/ET/STIM1(67),DTIM1(67),DTIM2(67),UPTIM(67)
      COMMON/LIMITS/MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      COMMON/ST15/LATST,CMNADD,IDATA
      COMMON/LOOPL/LDIST(67)
      COMMON/CLMN/DETECT(503),NDETNC
      COMMON/CTRL7/ERRCNT(18),DFRED
      COMMON/CTRL9/MCNT(10)
C
      INTEGER TDN0,TDN1,TDN2,TDN3
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER VOL15,OCC15,DVOL15,UVOL15,DOCC15,
     *   UOCC15,VSPEED,LVC15,LTCONT
      INTEGER STIM1,DTIM1,DTIM2,UPTIM
      INTEGER MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      INTEGER LATST,CMNADD,IDATA
      INTEGER UWAIT
      INTEGER LDIST
      INTEGER DETECT, NDETNC
      INTEGER ERRCNT,DFRED
      INTEGER MCNT
C
      INTEGER IN,WEAVE,SPEED,LENGTH,MAXET1,TDS
C
C
C PROCESS SINGLE-LOOP DATA
C
      IC=0
      DO 100 I=1,STOT ;NO OF. [illegible]
      IF(I.EQ.STOT)IC=204 ;SET [illegible]
      IN=IC+2
      IF((IN.EQ.18.OR.IN.EQ.29.OR.IN.EQ.40.OR.IN.EQ.51))IC=IC+1
      [illegible]
      IF(IN.EQ.96)IC=IC+1
      CALL SLCHEK(DETECT(IN),DETECT(IN+1),STIM1(I),WEAVE,I)
      VOL15(I)=VOL15(I)+DETECT(IN)
      OCC15(I)=OCC15(I)+DETECT(IN+1)
100   CONTINUE
C
C PROCESS DOUBLE-LOOP DATA
C
      IJ=STOT
      IC=102
      DO 200 I=1,DTOT
      IN=IC+5
      IF(IN.EQ.269)IN=IN+5 ;DUMMY
      IF((IN.EQ.125.OR.IN.EQ.141.OR.IN.EQ.157.OR.IN.EQ.173))IC=IC+1
      IF((IN.EQ.189.OR.IN.EQ.205.OR.IN.EQ.221.OR.IN.EQ.237))IC=IC+1
      [illegible]
      IF((IN.EQ.327.OR.IN.EQ.343.OR.IN.EQ.359.OR.IN.EQ.375))IC=IC+1
      IF((IN.EQ.391.OR.IN.EQ.407.OR.IN.EQ.423.OR.IN.EQ.439))IC=IC+1
      IF(IC.EQ.455)IC=IC+1
      IJ=IJ+1
      WEAVE=0
      LATST=0
      UWAIT=0
      [illegible]
      CALL DLCHEK(DETECT(IN+2),DETECT(IN+3),DETECT(IN+4),DTIM1(I),
     *DTIM2(I),WEAVE,UWAIT,I)
      DVOL15(I)=DVOL15(I)+DETECT(IN+2) ;ACC D. VOL
      DOCC15(I)=DOCC15(I)+DETECT(IN+3) ;ACC D. OCC
      UVOL15(I)=UVOL15(I)+DETECT(IN) ;ACC UP VOL
      UOCC15(I)=UOCC15(I)+DETECT(IN+1) ;ACC UP OCC
      IF(LATST.NE.0)GOTO 220 ;IGNORE SPEED & LENGTH CALCULATIONS
      [illegible] TO COMPUTE SPEED & LENG
      DTIM2(I)=DTIM2(I)/DETECT(IN+2) ;AVG DL TIME 2
      IF(DTIM2(I).LT.IMIN)DTIM2(I)=IMIN
      SPEED=LDIST(I)*25/DTIM2(I) ;SPEED IN M/SEC*10
      IF(SPEED.LT.0)SPEED=10 ;1M/S
      IF(SPEED.GT.MAXSPD)SPEED=MAXSPD
      LENGTH=STDLN ;SET DEFAULT IF IS A SHORT VEH(M*10)
      DTIM1(I)=DTIM1(I)/DETECT(IN+2) ;AVG DL TIME 1
      [illegible]
      IF(DTIM1(I).LE.MAXET1)LENGTH=(SPEED*DTIM1(I))/25-10
      IF(LENGTH.GT.STDLN)LVC15(I)=LVC15(I)+DETECT(IN+2)
      VSPEED(I)=(S1*VSPEED(I)+S2*SPEED)/10 ;P/S*10
220   DTIM1(I)=0
```

```
      DTIM2(I)=0
200   CONTINUE
C
      DO 300 I=1,NRAMPS
      IF(I.EQ.2.OR.I.EQ.4)GOTO 300
      IDX=NCNT(I)
      IF(IDX.EQ.6.OR.IDX.EQ.8)GOTO 310 ;2 RAMPS/CONTROLLER
      LTCOND(I)=LTCOND(I).OR.DETECT(IDX)
      GOTO 300
310   K=I ;N/B RAMP
      IF(ITEST(DETECT(IDX),6))K=I+1 ;S/B RAMP
      LTCOND(K)=LTCOND(K).OR.DETECT(IDX)
300   CONTINUE
C
C     INCREMENT 30 SEC. COUNTER
C
      IDATA=IDATA+1
C
C     IF 1 SEC. ELAPSED, THEN READY TASK 'ALMSGN'
C
      IF (.NOT. (IDATA.EQ.1)) GO TO 410
      CALL XMT (TSK3, 1, $405)
      GO TO 999
405   TYPE "IN SUBR 'DATAIN', XMT ERROR: TSK3 =", TSK3
      GO TO 999
C
C     IF 30 SEC. ELAPSED, THEN READY TASK 'TRANSF'
C
410   IF (.NOT. (MOD(IDATA,UFREQ).EQ.0)) GO TO 999
      IF (IDATA.EQ.2*UFREQ)
     *   IDATA=0
      CALL NXFER
      CALL XMT (TSK1, 1, $415)
      GO TO 999
415   TYPE "IN SUBR 'DATAIN', XMT ERROR: TSK1 =", TSK1
      GO TO 999
C
C     END OF ROUTINE
C
999   RETURN
      END
      OVERLAY ODATSTN
      SUBROUTINE DATSTN
      COMMON/NREC/RL305,NANEC,KINC,KSM,NALREC,KDISP
      COMMON/NRDAY/NMB(3),NOT(3)
      COMMON/INCA/INCTV(84),INCREC(21),INCFLAG(21),INCSTAT(21)
C
      COMMON/INCD/INCPNG(21),INCDT,INCVT,INCST
C
      INTEGER RL305,NMB,NOT,INCTV,INCREC,INCFLAG,INCSTAT
      INTEGER INCPNG,INCDT,INCVT,INCST
      INTEGER DATAX(264)
      INTEGER STN,I,J,IREC
      INTEGER DT308(768),REC30
      INTEGER DNOCC,DNVOL,DNSPD
C
C CREATE DISK FILE "DATAX" TO EXCHANGE DATA WITH CLOU ROUTINE
C
      CALL CFILW('DATAX',2,IER)
C
C
C UTILIZE INCSTAT ARRAY TO DETERMINE IF/AND WHERE AN INCIDENT OCCURRED
C
      STN=0
20    STN=STN+1
        IF(STN.GT.6)GOTO 999
        IF(STN.EQ.3)GOTO 20;                       STATION THREE HAS NO DOWNSTREAM STATION
C
C DETERMINE WHETHER OR NOT A NEW INCIDENT HAS BEEN DETECTED
C
        IF(INCSTAT(STN).EQ.0)INCPNG(STN)=0;        EITHER NO NEW INCIDENT OR AN INCIDENT HAS CLEARED
          IF(INCSTAT(STN).EQ.INCPNG(STN))GOTO 20;  NO NEW INCIDENT
C
C OBTAIN PRECEEDING 1 MIN. LOGICAL RECORD OF DATA30 FILE
C
        REC30=RL305-36
        I=1
        CALL OPEN(3,'DATA30',2,IER,128)
61      CALL RDRW(3,REC30,DT308,12,IER)
C
C EXTRACT , LANE OCCUPANCY ,VOLUME ,AND SPEED DATA
C
        DNOCC=341;                                 LAST ELEMENT - DOWNSTREAM OCCUPANCY
        DNVOL=207;                                 LAST ELEMENT - DOWNSTREAM VOLUME
        DNSPD=675;                                 LAST ELEMENT - LANE SPEEDS
        DO 22 I=1,66
C
C EXTRACT LANE OCCUPANCY DATA - STORE IN DATAX(1-66)
C
          IF(DNOCC.NE.323)GOTO 23                  IGNORE FOURTH LANE OF STATION SIX
            DNOCC=DNOCC-1;
C
23        DATAX(I)=DT308(DNOCC)                     IN THE EVENT OF DOWNSTREAM DETECTOR ERROR USE
          IF(DATAX(I).LT.0)DATAX(I)=DT308(DNOCC+67);  UPSTREAM DETECTOR DATA
C                                                   DECREMENT INDEX
          DNOCC=DNOCC-1;
C
C EXTRACT LANE VOL. DATA - STORE IN DATAX(67-132)
C
          IF(DNVOL.NE.189)GOTO 24
            DNVOL=DNVOL-1
C
24        DATAX(66+I)=DT308(DNVOL)
          IF(DATAX(66+I).LT.0)DATAX(66+I)=DT308(DNVOL+67)
C
          DNVOL=DNVOL-1
C
C EXTRACT LANE SPEED DATA - STORE IN DATAX(133-198)
C
          IF(DNSPD.EQ.457)DNSPD=DNSPD-1
          DATAX(132+I)=DT308(DNSPD)
C
          DNSPD=DNSPD-1
C
22        CONTINUE
C
C ISOLATE AVERAGE STN. OCC.VOL.SPD DATA
C
        DO 25 J=1,22
          DATAX(198+J)=DT308(542+J)
          DATAX(220+J)=DT308(608+J)
          DATAX(242+J)=DT308(674+J)
25        CONTINUE
C
C WRITE OCC. , VOL. , SPD. DATA TO DISK FILE "DATAX" FOR USE IN CLOU
```

```
      IREC=(I-1)*5
      CALL OPEN(8,'DATAX',2,IER,128)
      CALL WRTR(8,IREC,DATAX,5,IER)
      CALL CLOSE(8,IER)
C
C REPEAT PROCEDURE WITH PREVIOUS TIME SLICE
C USE LOGICAL RECORD OF "DATA30" FROM 30 SEC. EARLIER
C
      I=I+1
      REC30=REC30-12
      IF(I.LE.3)GOTO 41
      CALL CLOSE(3,IER)
C
999   RETURN
      END
```

```
C
      OVERLAY UDETECTION
      SUBROUTINE DETECTON
C
C     FUNCTIONS TO DISPLAY MAINLINE DETECTOR STATIONS WHITE ON BLACK ----- IF
C        AN ACCIDENT IS DETECTED AT A STATION THE DISPLAY IS BLACK ON WHITE
C     LACC = A LOGICAL VARIABLE TO INDICATE THE ACCIDENT STATUS OF THE LAST STA.
C
      COMMON/INCID/ACC,ALOC,AFLG
      COMMON/SCREEN/SCHN,PFILE,NVEW,NNTEMP
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      LOGICAL ACC(22),LACC,AFLG
      INTEGER SCHN,PFILE,NVEW,NNTEMP,ALOC(3),IDATA(39)
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
C
C
C
      WRITE(SCHN,50)
50    FORMAT(" ",*ICOL C0 C7")
      LACC=.FALSE.
      DO 100 I=1,NSTNS
      IF(ACC(I) .AND. LACC) GOTO 75
      IF(.NOT. ACC(I) .AND. LACC) WRITE(SCHN,250)
      IF(ACC(I) .AND. .NOT. LACC) WRITE(SCHN,200)
75    READ(PFILE,10) (IDATA(J),J=1,39)
      WRITE(SCHN,35) (IDATA(J),J=1,39)
      LACC=ACC(I)
100   CONTINUE
C
10    FORMAT(39A2)
35    FORMAT(" ",39A2)
200   FORMAT(" ",*ICOL C7 C0")
250   FORMAT(" ",*ICOL C0 C7")
C
      RETURN
      END
      OVERLAY ODIS1
      SUBROUTINE DIS1
C
      DIMENSION QEWDYN(512)
      DIMENSION IDNO(10)
      DIMENSION VOL15A(2),OCC15A(2)
      DIMENSION PDVOL(10),PDOCC(10),DDVOL(10),DDOCC(10)
      DIMENSION QDVOL(13),QDOCC(13),EDVOL(11),EDOCC(11)
      DIMENSION DNVOL(67),DNOCC(67),LNSPD(67)
C
      COMMON/WKDAY/HMS(3),MDY(3)
      COMMON/U30/DATA30(718)
      COMMON/MREC/KL30S,NMREC,KINC,K5M,NALREC,KDISP
      COMMON/TCNT/COUNT,KCNT5M,IMIN5,KCOUNT
      COMMON/GFOM/STOT,OTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/NDET/NPD,NOD,NED,NWD
C
      INTEGER QEWDYN
      INTEGER IDNO
      INTEGER VOL15A,OCC15A
      INTEGER PDVOL,PDOCC,DDVOL,DDOCC
      INTEGER QDVOL,QDOCC,EDVOL,EDOCC
      INTEGER DNVOL,DNOCC,LNSPD
      INTEGER HMS,MDY
      INTEGER DATA30
      INTEGER KL30S,NMREC,KINC,K5M,NALREC,KDISP
      INTEGER COUNT,KCNT5M,IMIN5,KCOUNT
      INTEGER STOT,OTOT,NRAMPS,NSTNS,CAP
      INTEGER NPD,NOD,NED,NWD
      INTEGER J,IJ1,IJ2,NUQ,NOE,GOOD1
      INTEGER K,K1,K2,K3,K4,IER
C
      EQUIVALENCE (VOL15A(1),DATA30(93)),(OCC15A(1),DATA30(130))
      EQUIVALENCE (PDVOL(1),DATA30(49)),(DDVOL(1),DATA30(59))
      EQUIVALENCE (EDVOL(1),DATA30(69)),(QDVOL(1),DATA30(80))
      EQUIVALENCE (PDOCC(1),DATA30(95)),(DDOCC(1),DATA30(105))
      EQUIVALENCE (EDOCC(1),DATA30(115)),(QDOCC(1),DATA30(126))
      EQUIVALENCE (DNVOL(1),DATA30(141)),(DNOCC(1),DATA30(275))
      EQUIVALENCE (LNSPD(1),DATA30(409))
C
C THIS PROGRAM UPDATES 1-MIN RAMP DET AND MAINLINE LANE DATA
C FOR THE GRAPHIC FILE
C
C NO OF D-DET(NSD),NO OF EXIT DET(LSD)
C
      IDNO(1)=10
      IDNO(2)=12
      IDNO(3)=11
      IDNO(4)=11
      IDNO(5)=10
      IDNO(6)=21
      IDNO(7)=21
      IDNO(8)=10
      IDNO(9)=12
      IDNO(10)=22
C
C     OPEN FILE
C
      CALL OPEN (3, 'QEWDYN', 2, IER, 128)
C
C     CHECK IF 30 SEC. OR 60 SEC. CALL
C
      CALL FGTIM (IHH, IMM, ISS)
X     WRITE (10, 333) IHH, IMM, ISS
X333  FORMAT (' IN SUBR 'DIS1', QEWDYN TIME = ', 3I5)
      KCOUNT = KCOUNT + 1
      IF (.NOT. (MOD(KCOUNT,2) .EQ. 1))  GO TO 20
C
C       THEN 30 SEC. CALL => RE-INITIALIZE QEW BUFFER TO ZERO
C
          DO 10  I = 1, 512
10            QEWDYN(I) = 0
          GO TO 30
C
C       ELSE 60 SEC. CALL => READ EXISTING QEW BUFFER
C
20        CONTINUE
          CALL READR (3, KDISP, QEWDYN(1), 8, IER)
          GO TO 30
C
C       END IF
C
30    CONTINUE
C
```

```
C UPDATE DATE AND TIME INFORMATION
C
      QEWDYN(1)=MDY(3)
      QEWDYN(2)=MDY(1)
      QEWDYN(3)=MDY(2)
      QEWDYN(4)=HMS(1)
      QEWDYN(5)=HMS(2)
      QEWDYN(6)=HMS(3)
C
C
C STN15A VOL AND OCC
C
      J=7 !BEG POINTER FOR TEMP WORKING BUFFER TO GET 30-SEC DATA
      QEWDYN(J)=VOL15A(1)+VOL15A(2) !STN15A VOL
      QEWDYN(J+1)=OCC15A(1)+OCC15A(2) !STN15A OCC IN PULSES
      IF(VOL15A(1).GE.0.AND.VOL15A(2).GE.0)GOTO 40!BOTH DETS OK
      IF(VOL15A(1).GE.0.OR.VOL15A(2).GE.0)GOTO 35 !ONLY 1 DET WORKING
      QEWDYN(J)=-1
      QEWDYN(J+1)=-1
      GOTO 40
35    GOOD1=2
      IF(VOL15A(2).LT.0)GOOD1=1
      QEWDYN(J)=VOL15A(GOOD1)*2
      QEWDYN(J+1)=OCC15A(GOOD1)*2
C
C RAMP DETECTOR 30-SEC VOL AND OCC
C
40    J=J+2
      IJ1=0
      IJ2=0
      DO 50 I=1,NRAMPR
      NOQ=IDNO(I)/10 !NO OF Q-DET FOR EACH RAMP
      NOE=MOD(IDNO(I),10) !NO OF EXIT DET FOR EACH RAMP
      QEWDYN(J)=PDVOL(I) !PASSAGE DET VOL
      QEWDYN(J+1)=PDOCC(I) !PASSAGE DET OCC
      QEWDYN(J+2)=DDVOL(I) !DEMAND DET VOL
      QEWDYN(J+3)=DDOCC(I) !DEMAND DET OCC
      J=J+4
      DO 52 K=1,NOQ
      IJ1=IJ1+1 !Q-DET INDEX
      QEWDYN(J)=QDVOL(IJ1)
      QEWDYN(J+1)=QDOCC(IJ1)
      J=J+2
52    CONTINUE
      IF(NOE.EQ.0)GOTO 50
      DO 54 K=1,NOE
      IJ2=IJ2+1 !EXIT DET INDEX
      QEWDYN(J)=EDVOL(IJ2)
      QEWDYN(J+1)=EDOCC(IJ2)
      J=J+2
54    CONTINUE
50    CONTINUE
C
      CALL WRITR(3,KDISP,QEWDYN,8,IER)
      CALL CLOSE(3,IER)
C
      RETURN
      END
      OVERLAY ODIS2
      SUBROUTINE DIS2
C
      DIMENSION OMPREC(7)
      DIMENSION QEWDYN(512),CYCLES(50)
      DIMENSION SSPD(22),SVOLIM(22),SOCCIM(22)
C
      COMMON/SHRP/NWRDS,BUFSIZ,NMBGTX,NMBGRX,MTLBGN(2)
      COMMON/QU2/QWAIT(10),QLNTH(10)
      COMMON/D30/DATA30(718)
      COMMON/NREC/KLSUB,NWREC,KINC,KSM,NALREC,KDISP
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/QOC/QOCC(13)
      COMMON/INCR/INCTV(84),INCREC(21),INCFLAG(21),INCSTAT(21)
C
      INTEGER OMPREC
      INTEGER QEWDYN,CYCLES
      INTEGER SSPD,SVOLIM,SOCCIM
      INTEGER NWRDS,BUFSIZ,NMBGTX,NMBGRX,MTLBGN
      INTEGER QWAIT,QLNTH
      INTEGER DATA30
      INTEGER KLSUB,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER CYCOVR,RAMCYC
      INTEGER QOCC
      INTEGER INCTV,INCREC,INCFLAG,INCSTAT
C
      EQUIVALENCE (SOCCIM(1),DATA30(587)),(SVOLIM(1),DATA30(609))
      EQUIVALENCE (SSPD(1),DATA30(675))
C
C THIS PROGRAM INTEGRATES RAMP CONTROL DATA AND MAINLINE STN 1-MIN DATA
C TO THE 1-MIN DATA CREATED IN DIS1.FR TO COMPLETE THE GRAPHIC FILE
C
C OPEN FILES
      CALL OPEN(3,"CYCLES",2,IER,100) ;CYCLE LNTH TABLE
      CALL READR(3,1,CYCLES(1),1,IER)
      CALL CLOSE(3,IER)
C
      CALL OPEN(3,"QEWDYN",2,IER,128)
      CALL READR(3,KDISP,QEWDYN(1),8,IER)
C
C UPDATE MAINLINE CONTROL MODE,CYCLE LENGTH,RAMP WAITING TIME
C AND SMOOTHED Q-OCCUPANCY
C
      L=0
      DO 10 I=1,NRAMPS
      J=(I-1)*5+7
      QEWDYN(J)=MODCYC(I)
      QEWDYN(J+2)=QWAIT(I)
      QEWDYN(J+4)=0 ;RESERVED
      QEWDYN(J+1)=0
      K=(I-1)*MAXCYC+RAMCYC(I)
      IF(RAMCYC(I).GT.0.AND.RAMCYC(I).LE.MAXCYC)QEWDYN(J+1)=CYCLES(K)
      IF(RAMCYC(I).EQ.0)QEWDYN(J+1)=-999
      L=L+1
      IF(I.EQ.4.OR.I.EQ.7.OR.I.EQ.10)GOTO 12 ;RAMPS WITH 2 Q-DETS
      QEWDYN(J+3)=QOCC(L) ;SMOOTHED Q-OCC
      IF(QOCC(L).GT.0)QEWDYN(J+3)=QOCC(L)*2/15 ;IN %
      GOTO 10
12    QEWDYN(J+3)=MAX0(QOCC(L),QOCC(L+1))
      IF(QEWDYN(J+3).GT.0)QEWDYN(J+3)=QEWDYN(J+3)*2/15 ;IN %
      L=L+1
10    CONTINUE
C
C CAPACITY,SPEED,VOL,OCC AND INCIDENT STATUS
C
      DO 20 I=1,NSTNS
      J=(I-1)*5+NRAMPS*5+7
      K=NSTNS-I+1 ;STN NO
      QEWDYN(J)=CAP(K) ;CAPACITY
      QEWDYN(J+1)=SSPD(K)*18/5 ;SPEED IN KM/H
      IF(SSPD(K).LT.0)QEWDYN(J+1)=-1
      QEWDYN(J+2)=SVOLIM(K) ;1-MIN STN VOL
      QEWDYN(J+3)=SOCCIM(K)/45 ;1-MIN STN OCC IN %
      IF(K.EQ.6)QEWDYN(J+3)=SOCCIM(K)/60 ;STN #6 AN EXTRA (FOUR LANES TOTAL) LANE
      IF(SOCCIM(K).LT.0)QEWDYN(J+3)=-1
      QEWDYN(J+4)=0 ;INC STATUS CALCULATIONS
      IF(K.EQ.3.OR.K.EQ.22)GOTO 20 ;STNS 3 AND 22 HAVE NO DOWNSTR. STN
      IF(K.GE.4.AND.INCSTAT(K).GT.0)QEWDYN(J+4)=1 ;DOWNSTR NO
      IF(K.GE.3)GOTO 20
      IF(K.EQ.2.AND.INCSTAT(K+1).GT.0)QEWDYN(J+4)=1 ;FOR STN 2
      IF(K.EQ.1.AND.INCOTAT(K).GT.0)QEWDYN(J+3)=1
      IF(K.EQ.1.AND.INCRTAT(K+1).GT.0)QEWDYN(J+4)=1
20    CONTINUE
C
      DO 50 I=1,24
      QEWDYN(457+I)=-1
50    CONTINUE
C
X     CALL FATIM (IHH, IMM, ISS)
X     WRITE (12, 333) IHH, IMM, ISS
X333  FORMAT (" IN SUBR 'DISP', WRITE TIME = ", 3I5 /)
      CALL WRITR(3.KDISP.QEWDYN(1).8.IER)
      CALL CLOSE(3.IER)
      KDISP=KDISP+8
C
C
C
      CALL OPEN(3,"DMPREC",2,IER,14)
      DMPREC(1)=KL308
      DMPREC(2)=NWREC
      DMPREC(3)=KINC
      DMPREC(4)=KSM
      DMPREC(5)=NALREC
      DMPREC(6)=KDISP
      DMPREC(7)=[illegible]
      CALL WRITR(3,1,DMPREC(1),1,IER)
      CALL CLOSE(3.IER)
C
      RETURN
      END
      OVERLAY MDISMAP
      SUBROUTINE DISMAP
C
C
C FUNCTIONS - READS POLYGON FILE TO OBTAIN INITIAL ELEMENT(+9) FOR OCCUPANCY
C                ON PASSING,CENTRE AND DRIVING LANES.
C             - READS THE NUMBER OF MAINLINE STATIONS IN THE INTERCHANGE MAP.
C             - DISPLAYS HEADING,DATE AND TIME FOR ALL INTERCHANGE MAPS
C
C
C          IODL,IOCL,IOPL,ISDL,ISCL,ISPL ARE VARIABLE NAMES FOR THE
C          WORDS CONTAINING THE OCCUPANCY & SPEED OF THE DRIVING,
C          CENTRE AND PASSING LANES.
C
C        I =  THE DETECTOR STATION NUMBER
C        NOP = THE NUMBER OF DETECTOR POLYGONS IN THE FILE
C        NSTA = # OF MAINLINE STATIONS IN DISPLAY
C
      COMMON/GRAPH/MAPNUM.COLOR
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN.PFILE.RQEW.RWTEMP
      COMMON/OPTA/ISDL.ISCL.ISPL
      COMMON/OPTB/IODL.IOCL.IOPL
      COMMON/EXTRA/WINSTON
      COMMON/WORDS/NSTA.NCON.IS.IV.IO.RO.RV.TXT.K.KI
      COMMON/OCCHN/OCC.RMETER
      INTEGER GCHN.PFILE.RQEW.RWTEMP
      INTEGER QEW(512).COLOR.MAPNUM.IDATA(39)
      LOGICAL WINSTON
      INTEGER NSTA.NCON.IS.IV.IO.RO.RV.TXT(6).K.KI
      LOGICAL RMETER(10).OCC
      WINSTON=.FALSE.
C
C     READ DATA FILE TO OBTAIN INITIAL SUBSCRIPTS (+9) FOR OCCUPANCY
C     ON THE DRIVING.CENTRE AND PASSING LANES
C
      READ(PFILE.45)IODL.IOCL.IOPL
45    FORMAT(3(I3.1X))
      ISDL=IODL+1
      ISCL=IOCL+1
      ISPL=IOPL+1
C
C     READ FILE TO DETERMINE # OF MAINLINE STATIONS IN FILE
C
      READ(PFILE.50)NSTA
50    FORMAT(I1)
C
C     READ AND PLACE DISPLAY HEADING(WITH COLOUR CRITERIA)
C                  -DATE AND
C                  -TIME
C
      READ(PFILE:18) (IDATA(J):J=1:39)
      WRITE(GCHN.35) (IDATA(J).J=1.39)
C
C     BECAUSE OF THE CONTENT OF MISS. RD. THE DATE AND TIME MUST BE PLACED
C     ELSEWHERE WITHIN THE DISPLAY
C
      IF(MAPNUM .EQ. 5) GOTO 75
C
C     IF COLOUR CRITERIA IS OCCUPANCY : PLACE "(OCC)"
C
      IF(OCC) WRITE(GCHN.55)
55    FORMAT(" "."IVEC 100 423ISTR/(OCC)/")
C
C     IF COLOUR CRITERIA IS SPEED : PLACE "(SPD)"
C
      IF (.NOT. OCC) WRITE(GCHN.60)
60    FORMAT(" "."IVEC 100 423ISTR/(SPD)/")
      WRITE(GCHN.65)(QEW(I).I=1.3)
65    FORMAT(" "."IVEC 146 423ISTR/DATE:".3I3."/")
      WRITE(GCHN.70)(QEW(I).I=4.6)
70    FORMAT(" "."IVEC 146 407ISTR/TIME:".3I3."/")
      RETURN
C
C     PLACEMENT OF DATE AND TIME FOR MISS. RD.
C
75    IF(OCC) WRITE(GCHN.90)
      IF(.NOT. OCC) WRITE(GCHN.95)
90    FORMAT(" "."IVEC 145 422ISTR/(OCC)/")
95    FORMAT(" "."IVEC 145 422ISTR/(SPD)/")
      WRITE(GCHN.80) (QEW(I).I=1.3)
      WRITE(GCHN.85) (QEW(I).I=4.6)
80    FORMAT(" "."IVEC 10 151ISTR/DATE:"3I3."/")
85    FORMAT(" "."IVEC 10 129ISTR/TIME:"3I3."/")
C
```

```
10      FORMAT(39A2)
35      FORMAT(' ',39A2)
        RETURN
        END
```

```
      SUBROUTINE DLCHEK(VOL,ET1,ET2,AEL1,AEL2,NVE,TWOSEC,IJ)
      COMMON/LIMITS/MAXOCC,MAXSPD,IMIN,MAXVEH,STOLN,S1,S2
C
      COMMON/STIS/LATST,ERRARD
      COMMON/RANGE/UPRNGE(113,2),DNRNGE(67,3)
      INTEGER MAXOCC,MAXSPD,IMIN,MAXVEH,STOLN,S1,S2
C
      INTEGER LATST,ERRARD
      INTEGER UPRNGE,DNRNGE
      INTEGER VOL,ET1,ET2,AEL1,AEL2,NVE,TWOSEC,IJ,IROW
C
C
C
      IROW=1
      IF(VOL.GT.MAXVEH.OR.VOL.LT.0)GOTO 120
      IF(ET1.LT.0.OR.ET2.LT.0)GOTO 140
      IF(ET1.GT.MAXOCC)ET1=MAXOCC
      IF(ET2.GT.MAXOCC)ET2=MAXOCC
      IF(NVE.EQ.1.AND.VOL.EQ.0.AND.ET1.EQ.0)GOTO 125
      IF(NVE.EQ.1.AND.VOL.GT.0.AND.ET1.GT.0.AND.ET2.EQ.0)GOTO 125
      IF(NVE.EQ.0.AND.VOL.EQ.0.AND.ET1.EQ.0)TWOSEC=TWOSEC+1
      IF(VOL.GT.0.OR.ET1.GT.0)TWOSEC=0
      IF(TWOSEC.GE.2)GOTO 125
      AEL1=AEL1+ET1 ;ACC EL TIME 1
      IF(VOL.EQ.0)GOTO 130
      IROW=2
      IF(AEL1.LT.(VOL*IMIN))GOTO 120
130   AEL2=AEL2+ET2 ;ACC EL TIME 2
      IF(VOL.EQ.0)GO TO 999
      IF(AEL2.GE.(VOL*IMIN))GO TO 999
      LATST=1 ;FLAG TO IND NOT TO COMPUTE SPEED
      IROW=3
120   DNRNGE(IJ,IROW)=DNRNGE(IJ,IROW)+1
140   ERRARD=ERRARD+1 ;INC ERROR COUNTS
      IF(LATST.EQ.1)GOTO 135 ;DO NOT SET DNSTREAM VALUES TO 0
125   VOL=0
      ET1=0
      AEL1=0
135   ET2=0
      AEL2=0
      TWOSEC=0
C
999   RETURN
      END
```

```
C
      OVERLAY ODYS1
      SUBROUTINE DYS1
C
      DIMENSION QEWDYN(512)
      DIMENSION IUNO(10)
      DIMENSION VOL15A(2),OCC15A(2)
      DIMENSION PDVOL(10),PDOCC(10),DDVOL(10),DDOCC(10)
      DIMENSION QDVOL(13),QDOCC(13),EDVOL(11),EDOCC(11)
      DIMENSION DNVOL(67),DNOCC(67),LNSPD(67)
C
      COMMON/WKDAY/HMS(3),MDY(3)
      COMMON/D30/DATA30(718)
      COMMON/NREC/KL30B,NWREC,KINC,K5M,NALREC,KDISP
      COMMON/TCNT/COUNT,KCNT5M,IMINS,KCOUNT
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/NDET/NPD,NOD,NED,NQD
C
      INTEGER QEWDYN
      INTEGER IUNO
      INTEGER VOL15A,OCC15A
      INTEGER PDVOL,PDOCC,DDVOL,DDOCC
      INTEGER QDVOL,QDOCC,EDVOL,EDOCC
      INTEGER DNVOL,DNOCC,LNSPD
      INTEGER HMS,MDY
      INTEGER DATA30
      INTEGER KL30B,NWREC,KINC,K5M,NALREC,KDISP
      INTEGER COUNT,KCNT5M,IMINS,KCOUNT
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER NPD,NOD,NED,NQD
      INTEGER J,IJ1,IJ2,NOQ,NOE,G0001
      INTEGER K,K1,K2,K3,K4,IEN
C
      EQUIVALENCE (VOL15A(1),DATA30(93)),(OCC15A(1),DATA30(139))
      EQUIVALENCE (PDVOL(1),DATA30(49)),(DDVOL(1),DATA30(59))
      EQUIVALENCE (EDVOL(1),DATA30(69)),(QDVOL(1),DATA30(80))
      EQUIVALENCE (PDOCC(1),DATA30(95)),(DDOCC(1),DATA30(105))
      EQUIVALENCE (EDOCC(1),DATA30(115)),(QDOCC(1),DATA30(126))
      EQUIVALENCE (DNVOL(1),DATA30(141)),(DNOCC(1),DATA30(275))
      EQUIVALENCE (LNSPD(1),DATA30(409))
C
C THIS PROGRAM UPDATES 1-MIN RAMP DET AND MAINLINE LANE DATA
C FOR THE GRAPHIC FILE
C
C OPEN FILE
      CALL OPEN(3,"QEWDYN",2,IER,128)
      CALL READR(3,KDISP,QEWDYN(1),8,IER)
C
      K1=NRAMPS*5+NSTNS*5+6
      K2=(NPD+NOD+NQD+NED)*2+2
      K3=K1+K2+1
C
      IF(MOD(KCOUNT,2).EQ.0)GOTO 90 ;END OF MINUTE
C
      DO 60 I=1,K2
      QEWDYN(K1+I)=QEWDYN(I+6)
60    QEWDYN(I+6)=0
C
C
C 30-SEC MAINLINE LANE DATA
C
      DO 70 I=1,DTOT
      QEWDYN(K3)=DNVOL(I)
      QEWDYN(K3+1)=DNOCC(I)
      QEWDYN(K3+2)=LNSPD(I)
      K3=K3+3
70    CONTINUE
C
C WRITE 30-SEC DATA TO FILE
C
      GOTO 900
C
C GET 1-MIN DATA
C
90    IF(QEWDYN(K1+1).LT.0.OR.QEWDYN(7).LT.0)GOTO 80
      QEWDYN(K1+1)=QEWDYN(K1+1)+QEWDYN(7) ;STN15A VOL
      QEWDYN(K1+2)=(QEWDYN(K1+2)+QEWDYN(8))/30 ;STN15A OCC IN %
      GOTO 200
80    QEWDYN(K1+1)=-1
      QEWDYN(K1+2)=-1
C 1-MIN RAMP DETECTOR VOL AND OCC
200   DO 300 I=3,K2
      IF(QEWDYN(K1+I).LT.0.OR.QEWDYN(I+6).LT.0)GOTO 310
      QEWDYN(K1+I)=QEWDYN(K1+I)+QEWDYN(I+6)
      K4=MOD(I,2)
      IF(K4.EQ.0)QEWDYN(K1+I)=QEWDYN(K1+I)/15 ;OCC IN %
      GOTO 300
510   QEWDYN(K1+1)=-1
300   CONTINUE
C
C  1-MIN MAINLINE LANE DATA(VOL.OCC.SPEED)
C
      DO 500 I=1.QTOT
      IF(QEWDYN(K3).LT.0.OR.DNVOL(I).LT.0)GOTO 510
      QEWDYN(K3)=QEWDYN(K3)+DNVOL(I)
      QEWDYN(K3+1)=(QEWDYN(K3+1)+DNOCC(I))/15 ;OCC IN %
      IF(QEWDYN(K3+2).LT.0.OR.LNSPD(I).LT.0)GOTO 520
      QEWDYN(K3+2)=(QEWDYN(K3+2)+LNSPD(I))*9/5 ;AVG SPD IN KM/H
      GOTO 530
510   QEWDYN(K3)=-1
      QEWDYN(K3+1)=-1
520   QEWDYN(K3+2)=-1
530   K3=K3+3
500   CONTINUE
C
900   CALL WRITR(3.KDISP.QEWDYN.8.IER)
      CALL CLOSE(3.IER)
C
      RETURN
      END
```

```
C
      OVERLAY OFACC
      SUBROUTINE FACC
C
C
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/INCID/ACC,ALOC,AFLG
C
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      LOGICAL ACC(22),AFLG
      INTEGER ALOC(3)
C
C     LOCATE FIRST 3 ACCIDENT LOCATIONS
C
      J=1
      DO 100 I=1,NSTNS
      IF(ACC(I)) ALOC(J)=I
      IF(ACC(I)) J=J+1
      IF(J .GE. 4) GO TO 999      ;RECORD FIRST 3 ACC LOCATIONS ONLY
100   CONTINUE
C
999   RETURN
      END
      OVERLAY OFLASH
      SUBROUTINE FLASH
C
C FUNCTION : TO COLLECT THE FIRST 3 ACCIDENT LOCATION POLYGONS AND WRITE THEM
C            INTO A TEMPORARY FILE WHICH IS LATER DISPLAYED USING "LOOP"
C
C
      COMMON/GEOM/STOT.DTOT.NRAMPS.NSTNS.CAP(22)
      COMMON/SCREEN/SCRN.PFILE.RQEW.RWTEMP
      COMMON/INCID/ACC.ALOC.AFLG
      COMMON/NACC/KCNT
      COMMON/WORDS/NSTA.NCON.IS.IV.IO.RO.RV.TXT.K.K1
C
      INTEGER STOT.DTOT.NRAMPS.NSTNS.CAP
      INTEGER NSTA.NCON.IS.IV.IO,RO.RV.TXT(6).K.K1
      INTEGER IDATA(39)
      INTEGER SCRN.PFILE.RQEW.RWTEMP.ALOC(3)
      LOGICAL ACC(22).AFLG
C
C     "TEMPFILE" WAS CREATED IN ROUTINE INITIALIZE.FR
C       OPEN FILE TO CONTAIN POLYGONS TO BE FLASHED
C
      CALL OPEN(RWTEMP."TEMPFILE".2.IER.78)
      REWIND PFILE
      KCNT=0
C
C     TEST FOR LOCATION OF FIRST ACCIDENT
C
      IAL = ALOC(1)
      IF ( (IAL.LT.1) .OR. (IAL.GT.3) ) GO TO 200
      GO TO (300.400.500). IAL
C
300   DO 325 I=1.2
325   READ(PFILE.18) (IDATA(L).L=1.39)
      GOTO 800
400   DO 425 I=1.4
425   READ(PFILE.18) (IDATA(L).L=1.39)
      GOTO 800
500   DO 510 I=1.5
510   READ(PFILE.18) (IDATA(L).L=1.39)
800   READ(PFILE.18) (IDATA(L).L=1.39)
18    FORMAT (39A2)
      WRITE(RWTEMP.35) (IDATA(L).L=1.39)
35    FORMAT (' '.39A2)
      KK=ISHFT(IDATA(39).8)+000046K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 800;CHECK FOR 4027 CONT.
      KCNT=KCNT+1
      DO 540 I=2.3
      IF(ALOC(I) .EQ. 2) GOTO 530
      IF(ALOC(I) .EQ. 3 .AND. ALOC(I-1) .EQ. 2) GOTO 530
      IF(ALOC(I) .EQ. 3 .AND. ALOC(I-1) .EQ. 1) GOTO 525
      GOTO 550
525   READ(PFILE.18) (IDATA(L).L=1.39)
530   READ(PFILE.18) (IDATA(L).L=1.39)
      WRITE(RWTEMP.35) (IDATA(L).L=1.39)
      KK=ISHFT(IDATA(39).8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 530
      KCNT=KCNT+1
540   CONTINUE
550   IF(KCNT .GE. 3) GOTO 1000        ;ONLY ALLOW FIRST 3 ACC LOCATIONS
      REWIND PFILE
200   DO 700 I=4.NSTNS
      IF(.NOT. ACC(I)) GOTO 700
      IF(KCNT .GE. 3) GOTO 1000
      NREC=I+1
      DO 600 J=1.NREC
      READ(PFILE.18) (IDATA(L).L=1.39)
600   CONTINUE
C
C     THIS EXTRA READ STMT AND CONTINUATION CHECK IS PROVIDED FOR
C     FUTURE POLYGONS WHICH MAY CONTAIN A 4027 CONTINUATION.
C
900   READ(PFILE.18) (IDATA(J).J=1.39)
      WRITE(RWTEMP.35) (IDATA(J).J=1.39)
      KK=ISHFT(IDATA(39).8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 900
      KCNT=KCNT+1
      REWIND PFILE
700   CONTINUE
C
1000  RETURN
      END
```

```
C
C     TASK GMAP
C
C     A ONE-MINUTE PERIODIC TRAFFIC STATUS DISPLAY TASK.
C       UTILIZING COLOUR GRAPHIC CRT TERMINAL.
C
C
C ALL THE DATA TO BE DISPLAYED IS CONTAINED IN A FILE OF TEKTRONIX 4027 COMMANDS
C THESE POLYGONS ARE COLOURED ACCORDING TO EITHER THE OCCUPANCY OR SPEED DATA.
C
      EXTERNAL OREQEW,OCHQUW,OACCIDENT,OUSPD,OOOCC
      EXTERNAL ORAMPS,ODETECTON,OSPD1,OSPD2,OIOCC,OIOC2
      EXTERNAL OLOOP,OHOX1,OHOX2,OHOX3,OPCROX
      EXTERNAL OFLASH,OFACC,ODISMAP,ORDISMAP
C
      COMMON/BCOM/STOT,OTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/LASTMAP/LMAP
      COMMON/GRAPH/MAPNUM,COLOR
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/SGM3/SGNSTAT(2),ALMFLG,KCNTD,P4FLG,AMODE
      COMMON/MD/MODEMT(299),MODEM(146),MODEM1(73),NUMMOM
      COMMON/SGM1/XMTBUF(45),RCVBUF(45),SGNADR(2)
      COMMON/SGM2/NWORDS,BUFSIZ,NMSGTX,NMSGRX,MTLSGN(2)
      COMMON/SCREEN/GCHN,PFILE,NNEW,NWTEMP
      COMMON/TSKSTN/TSK0,TSK1,TSK2,TSK3
      COMMON/INCID/ACC,ALOC,AFLG
      COMMON/OCCRM/OCC,RMETER
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      COMMON/OPTA/ISOL,ISCL,ISPL
      COMMON/OPTB/IOOL,IOCL,IOPL
C
      INTEGER STOT,OTOT,NRAMPS,NSTNS,CAP
      INTEGER LMAP
      INTEGER MAPNUM,COLOR
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER SGNSTAT,ALMFLG,KCNTD,P4FLG,AMODE
      INTEGER MODEMT,MODEM,MODEM1,NUMMOM
      INTEGER XMTBUF,RCVBUF,SGNADR
      INTEGER NWORDS,BUFSIZ,NMSGTX,NMSGRX,MTLSGN
      INTEGER GCHN,PFILE,NNEW,NWTEMP
      INTEGER TSK0,TSK1,TSK2,TSK3
      INTEGER ALOC(3)
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1
      INTEGER ISOL,ISCL,ISPL
      INTEGER IOOL,IOCL,IOPL
C
      LOGICAL ACC(22),AFLG
      LOGICAL OCC,RMETER(10)
C
C
C     INTERNAL VARIABLES:
C
      INTEGER I,IDATA(39),IER,MD2
C
C
C
C     TASK IS READIED BY TASK 'TRANSF'
C
600   CONTINUE
      CALL REC (TSK2, MD2)
      IF (MAPNUM .LT. 1)  GO TO 600
C
C     OPEN CHANNEL TO TEKTRONIX 4027 COLOUR GRAPHICS TERMINAL
C
      CALL OPEN (GCHN, 'STTO1', 5, IER, 2*39)
      IF (IER .NE. 1)  GO TO 600
C
C     READ WENDYN DATA FILE
C
      CALL FGTIM (IHH, IMM, ISS)
      WRITE (12, 1111) IHH, IMM, ISS
1111  FORMAT (' GMAP TIME =', 3I5)
      CALL FOVLY(1,OREQEW,0,IER)
      CALL REQEW
      CALL FUVRL(OREQEW,IER)
C
C     COLOUR MAP BASED ON OCCUPANCY OR SPEED ? SET C5 TO ORANGE
C
      IF(LMAP .EQ. MAPNUM) GO TO 550
      WRITE(GCHN,300)
300   FORMAT(" ","!WOR 33!SIGMA 1 33!MAP C5 140 50 75")
550   IF(COLOR .EQ. 1) OCC=.FALSE.     !IF TRUE, COLOUR BASED ON SPEED
C
C     CHECK FOR 0 OVERRIDE CONDITIONS ON RAMPS
```

```
C
      CALL FOVLD(1.OCHOOM.0.IER)
      CALL CHWOR
      CALL FOVRL(OCHWOM.IER)
C
C     SET ALL RAMPS IN METERING CONDITION
C
      DO 25 I=1.NRAMPS
      RMETER(I)=.TRUE.
25    CONTINUE
C
C     CHECK ON RAMP METERING
C
      DO 50 I=1.NRAMPS
      IF(METER(I) .EQ. 0) RMETER(I)=.FALSE.   !NO RAMP METERING
50    CONTINUE
C
C     CHECK FOR ACCIDENTS
C
      CALL FOVLD(1.OACCIDENT.0.IER)
      CALL ACCIDENT
      CALL FOVRL(OACCIDENT.IER)
C
C
C     OPEN POLYGON FILES ON CHANNEL 7(PFILE)
C
      LMAP=MAPNUM
      IF(MAPNUM .EQ. 1) CALL OPEN(PFILE."QEWPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 2) CALL OPEN(PFILE."FORDPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 3) CALL OPEN(PFILE."WINPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 4) CALL OPEN(PFILE."ERINPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 5) CALL OPEN(PFILE."MISSPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 6) CALL OPEN(PFILE."HWYPOLY".1.IER.2*39)
      IF(MAPNUM .EQ. 7) CALL OPEN(PFILE."CAMPOLY".1.IER.2*39)
      GO TO(1.2.2.2.2.2.2).MAPNUM
C
C
C *** M A I N   N E W   D I S P L A Y  ***
C
C     PLACE HEADING.DATE.TIME. & CHOOSE COLOUR OPTIONS
C
1     CALL FOVLD(1.ODSPD.0.IER)
      CALL DSPD
      CALL FOVRL(ODSPD.IER)
      IF(.NOT. UCC) GO TO 75
      CALL FOVLD(1.OQDCC.0.IER)
      CALL QDCC
      CALL FOVRL(OQDCC.IER)
C
C     DISPLAY BRIDGES BETWEEN STA 6 & 7 AND BETWEEN 20 & 21
C
75    DO 700 I=1.4
      READ(PFILE.10) (IDATA(J).J=1.39)
      WRITE(GCHN.35) (IDATA(J).J=1.39)
700   CONTINUE
C
C     DISPLAY VECTORED RAMPS
C
      CALL FOVLD(1.ORAMPS.0.IER)
      CALL RAMPS
      CALL FOVRL(ORAMPS.IER)
C
C     DISPLAY MAINLINE STATIONS
C
      CALL FOVLD(1.ODETECTOR.0.IER)
      CALL DETECTOR
      CALL FOVRL(ODETECTOR.IER)
C
C     DISPLAY HWY 2 & HWY 5 PCS BOXES
C
      CALL FOVLD(1.OPCBOX.0.IER)
      CALL PCBOX
      CALL FOVRL(OPCBOX.IER)
C
C     DISPLAY REST OF FILE (STREETS.ETC)
C
5     READ(PFILE.10.END=99) (IDATA(I).I=1.39)
10    FORMAT(39A2)
      WRITE(GCHN.35) (IDATA(I).I=1.39)
      GO TO 5
35    FORMAT(" ".39A2)
C
C     CALL FLASH IF AN ACCIDENT HAS BEEN DETECTED
C
```

```
        IF(.NOT. AFLG) GO TO 400        ;NO ACCIDENT DETECTED
C
C       GET LOCATIONS OF FIRST 5 ACCIDENTS
C
        CALL FOVLD(1,OFACC,0,IER)
        CALL FACC
        CALL FOVRL(OFACC,IER)
C
C       WRITE THE POLYGONS FOR THE ACCIDENT LOCATIONS INTO A TEMPORARY FILE
C
        CALL FOVLD(1,OFLASH,0,IER)
        CALL FLASH
        CALL FOVRL(OFLASH,IER)
C
C       FLASH POLYGONS BY LOOPING ON "TEMPFILE"
C
        CALL FOVLD(1,OLOOP,0,IER)
        CALL LOOP
        CALL FOVRL(OLOOP,IER)
C
C       CHECK TO DISPLAY MESSAGE ON MAIN OEW DISPLAY
C
400     CONTINUE
X       WRITE (10, 2222) (MODEMT(JJ), JJ=1,40)
X2222   FORMAT (" IN TASK 'GMAP', MODEMT = " / (1X, 10I6))
        IF(MODEMT(40) .LT. 0) WRITE(GCHN,12)
        IF(SGNSTAT(1) .GT. 0) WRITE(GCHN,11)
11      FORMAT(" ",<IVEC 40 0ISTN<SIGN1 OR SIGN2 MALFUNCTION<")
12      FORMAT(" ",<IVEC 320 0ISTN<SIGN MODEM MALFUNCTION<")
        IF(NMSGTX .EQ. 0) GO TO 300      ;DISPLAY NO CMS
C
C       DISPLAY CMS
C
        WRITE(GCHN,1000) (XMTBUF(I),I=3,13)
        WRITE(GCHN,2000) (XMTBUF(I),I=15,25)
1000    FORMAT(" ",<IVEC 440 211ISTR<",11A2,"<")
2000    FORMAT(" ",<IVEC 440 71ISTR<",11A2,"<")
        GO TO 300 .
C
C       I N T E R C H A N G E    D I S P L A Y S
C
C
C       INITIALIZE DL,CL,PL ELEMENTS
C       DETERMINE NUMBER OF MAINLINE STATIONS FOR DISPLAY
C       PLACE HEADING ,DATE,TIME
C
        CALL FOVLD(1,ODISMAP,0,IER)
        CALL DISMAP
        CALL FOVRL(ODISMAP,IER)
        IF(OCC) GO TO 150;              IF TRUE, COLOR BASED ON OCCUPANCY
C
C ***** COLOUR CRITERIA  S P E E D  *****
C
        DO 100 I=1,NSTA
        CALL FOVLD(1,OSPD1,0,IER)
        ISCL=ISCL-9
        ISPL=ISPL-9
        CALL SPD1
        CALL FOVRL(OSPD1,IER)
C
        CALL FOVLD(1,OSPD2,0,IER)
        ISDL=ISDL-9
        CALL SPD2
        CALL FOVRL(OSPD2,IER)
100     CONTINUE
C
        CALL FOVLD(1,ORAMPS,0,IER)
        CALL RAMPS
        CALL FOVRL(ORAMPS,IER)
C
        GO TO 200
C
C ***** COLOUR CRITERIA  O C C U P A N C Y  *****
C
150     CALL FOVLD(1,OIOCC,0,IER)
C
        DO 175 I=1,NSTA
        IODL=IODL-9
        IOCL=IOCL-9
        IOPL=IOPL-9
        CALL IOCC        ; SET COLOUR AND DRAW DL,CL,PL
175     CONTINUE
C
        CALL FOVRL(OIOCC,IER)
C
```

```
C     IF(MAPNUM .NE. 3) GO TO 100;           NOT DISPLAYING WINSTON
      CALL FOVLD(1.OIOC2.0.IER)
      CALL IOC2
      CALL FOVRL(OIOC2.IER)
C
C     DISPLAY INTERCHANGE RAMPS AS POLYGONS
C
100   CALL FOVLD(1.ORAMPS.0.IER)
      CALL RAMPS
      CALL FOVRL(ORAMPS.IER)
C
C     READ NUMBER OF RAMP CONTROLLERS;
C     INITIALIZE SPEED.VOLUME AND OCCUPANCY ELEMENTS;
C     DRAW GREEN BOXES
C
200   CALL FOVLD(1.OBOX1.0.IER)
      CALL BOX1
      CALL FOVRL(OBOX1.IER)
C
C     DISPLAY MAINLINE AND RAMP STATION DATA
C
      CALL FOVLD(1.OBOX2.0.IER)
      CALL BOX2
      CALL FOVRL(OBOX2.IER)
C
C     DISPLAY CYCLE TIME FOR ON RAMPS AND
C     STATION DATA FOR 14A ON MISS ROAD
C
      CALL FOVLD(1.OBOX3.0.IER)
      CALL BOX3
      CALL FOVRL(OBOX3.IER)
C
C     DISPLAY REST OF FILE
C             -MISCELLANEOUS POLYGONS
C             -DETECTOR STATIONS
C             -RAMP METERING ARROWS
C             -LEGEND.TEXTUAL DATA.NORTH POINT
C
      CALL FOVLD(1.ORDISMAP.0.IER)
      CALL RDISMAP
      CALL FOVRL(ORDISMAP.IER)
C
C     CLOSE CHANNELS
C
300   CONTINUE
      CALL CLOSE(PFILE.IER)
      CALL CLOSE(SCHN.IER)
C
C     END OF ROUTINE
C
      GO TO 600
      END
      OVERLAY OHEADING
      SUBROUTINE HEADING
C
C
      COMMON/NWEC/KL308,NWREC,KINC,KSM,NALREC,KDISP
      COMMON/MKDAY/HMS(3),MDY(3)
      COMMON/WPV/WEATHER(36)
      INTEGER KL308,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER HMS,MDY
      INTEGER WEATHER
C
      INTEGER IST(64),WEATH(10),PAVE(10),VISIB(10)
C
      EQUIVALENCE (WEATH(1),WEATHER(7)),(PAVE(1),WEATHER(17)),
     *      (VISIB(1),WEATHER(27))
C
C
C
      WRITE(10,300)
300   FORMAT (' ENTER CONDITIONS:')
      WRITE(10,310)
310   FORMAT ('    WEATHER     >',Z)
      READ(11,210)WEATH(2)
210   FORMAT(S16)
      WRITE(10,320)
320   FORMAT ('    PAVEMENT    >',Z)
      READ(11,210)PAVE(2)
      WRITE(10,330)
330   FORMAT ('    VISIBILITY  >',Z)
      READ(11,210)VISIB(2)
C
C     GET TODAY'S DATE
C
      CALL DATE (MDY(1), IER)
C
C     WAIT TILL EVEN MINUTE
C
130   CALL TIME (HMS(1), IER)
      IF(HMS(3).NE.00)GO TO 130
C
C     PRINT HEADING
C
      WRITE(10,120)
120   FORMAT (//1X,'* WELCOME TO Q.E.M. SURV. AND CONTROL PROJECT *',
     *       //1X,'*      COURTESY OF SYSTEMS RESEARCH - M.T.C.     *')
      WRITE(10,140) MDY(3),MDY(1),MDY(2),HMS
140   FORMAT (//1X,'* TO-DAY'S DATE: ',I2,"/",I2,"/",I2,
     *        6X,"TIME: ",I2,":",I2,":",I2," *")
C
C     SAVE DATE AND TIME IN 'WEATHER' FILE
C
      CALL OPEN(2,"WEATHER",3,IER,2*36)
      WEATHER(1)=MDY(3)
      WEATHER(2)=MDY(1)
      WEATHER(3)=MDY(2)
      DO 150 I=1,3
         WEATHER(I+3)=HMS(I)
150     CONTINUE
      CALL WRITR(2,NWREC,WEATHER(1),1,IER)
      CALL CLOSE(2,IER)
      NWREC=NWREC+1
C
      RETURN
      END
      OVERLAY DINCDET
      SUBROUTINE INCDET
      DIMENSION INCLOG(25)
      DIMENSION CSOCC(22),PSOCC(22),SOCCIM(22),SVOLIM(22)
      COMMON/INCA/INCTV(84),INCREC(21),INCFLAG(21),INCSTAT(21)
      COMMON/INCB/OCCDWI(21),OCCUWI(21),INCNT(21),INCONT,INCEX
      COMMON/INCC/INWMOL
      COMMON/INCE/CHCK(21),CMNI1(21),CMNJ(21),TEST3
      COMMON/NMDAY/NMS(3),MDY(3)
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/SECTL/DIST(20)
      COMMON/T18/TT(20)
      COMMON/D30/DATA30(718)
      COMMON/NWEC/RL30B,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER INCLOG
      INTEGER CSOCC,PSOCC,SOCCIM,SVOLIM
      INTEGER INCTV,INCREC,INCFLAG,INCSTAT
      INTEGER OCCDWI,OCCUWI,INCNT,INCONT,INCEX
      INTEGER INWMOL
      INTEGER NMS,MDY
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER DIST
      INTEGER TT
      INTEGER DATA30
      INTEGER RL30B,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER NSTNM1,I1,J,K,KK,IK
      INTEGER TEST1,TEST2,TEST4,OCCAVG,PSOD,IDIFF
      INTEGER MOL,IJ,IJ1,ITT,TSUM
      INTEGER CHCK,CMNJ,CMNI1,TEST3
      EQUIVALENCE (CSOCC(1),DATA30(543)),(PSOCC(1),DATA30(565)),
     *(SOCCIM(1),DATA30(587)),(SVOLIM(1),DATA30(609))
C
C
      NSTNM1=NSTNS-1
      DO 99 I=1,NSTNM1
      CHCK(I)=1
      KK=6*I-5
      I1=I !UPSTR. STN NO
      IF(I.EQ.2.OR.I.EQ.3)I1=I-1
      IF(CSOCC(I1).LT.0)GOTO 100 !UPSTR. STN FAIL
      J=I
      IF(I.EQ.1.AND.J.GE.4)GOTO 7
5     J=J+1 !DOWNSTR. STN NO.
10    IF(J.GT.NSTNS)GOTO 100
7     IF(I.EQ.3)GOTO 100 !STN3 HAS NO DOWNSTR STN
      IF(CSOCC(J).GE.0.AND.SOCCIM(J).GE.0)GOTO 15
      IF(I.EQ.1)J=J+1
      IF(I.EQ.1)GOTO 5
      GOTO 10
C
C     COMPUTE TEST1,2,3,4 STATISTICS
C
15    CMNI1(I)=I1
      CMNJ(I)=J
      TEST1=CSOCC(I1)*4/9 !U. STN OCC IN %*10
      PSOD=CSOCC(J)*4/9 !D.STN OCC IN %*10
      TEST2=TEST1-PSOD !TEST2 IN %*10
      TEST3=-999
      TEST1=(TEST1+5)/10  !TEST1 IN %,ROUND-OFF
      IF(TEST1.GT.0)TEST3=TEST2*10/TEST1 !TEST3 IN %
      OCCAVG=SOCCIM(J)*2/9;AVG D. OCC IN %*10
      IDIFF=OCCAVG-PSOD
      TEST4=999
      IF(IDIFF.LT.0)TEST4=-999
      PSOD=(PSOD+5)/10  !PSOD IN %
      TEST2=(TEST2+5)/10  !TEST2 IN %
      OCCAVG=(OCCAVG+5)/10 !OCCAVG IN %
      IF(OCCAVG.GT.0)TEST4=IDIFF*10/OCCAVG !TEST4 IN %
      K=4*I-3
      IF(INCFLAG(I).EQ.0) GOTO 13      !NO INCIDENT DETECTED
      IF(INCSTAT(I).EQ.0) GOTO 99      !INCIDENT NOT YET CONFIRMED
      GO TO 20
C
C CHECK INCIDENT OCCURRENCE
C
13    IF(TEST1.LT.INCTV(K))GOTO 100 !NO PREV INC
      IF(TEST2.LT.INCTV(K+1).OR.TEST3.LT.INCTV(K+2))GOTO 100! NO INC
      IF(TEST4.LT.INCTV(K+3))GOTO 100! NO INC
      INCFLAG(I)=1 !INC DETECTED
      OCCDWI(I)=OCCAVG !DOWNSTR OCC JUST BEFORE INC
      OCCUWI(I)=PSOCC(I1) !UPSTR OCC JUST BEFORE INC
      GOTO 99
20    IF(TEST3.GE.INCTV(K+2))GOTO 99 !INC STILL ON
      IF(PSOD.GE.OCCDWI(I).AND.OCCUWI(I).LT.0)GOTO 17 !INC CLEARED
      IF(PSOD.LT.OCCDWI(I).OR.CSOCC(I1).GT.OCCUWI(I))GOTO 99! INC STILL ON
```

```
         IF(VUUU.LE.0.AND.OCCOWI(I).EQ.0)GOTO 99 ;INC STILL ON
17       INCFLAG(I)=0 ;INC CLEANED
         OCCOWI(I)=0
         OCCUWI(I)=0
         INCONI=0
         GO TO 99
100      CHCK(I)=0
99       CONTINUE
         RETURN
         END
```

```
      OVERLAY DINITIALIZE
      SUBROUTINE INITIALIZE
C
C
      EXTERNAL COMCOM.TRANSF.ALMSGN.GMAP
C
      COMMON/SGN1/XMTBUF(45).RCVBUF(45).SGNADR(2)
      COMMON/SGN2/NWORDS.BUFSIZ.NMSGTX.NMSGRX.MTLSGN(2)
      COMMON/D30/DATA30(71A)
      COMMON/NREC/KL308.NWREC.KINC.KSM.NALRFC.KDISP
C
      INTEGER XMTBUF.RCVBUF.SGNADR
      INTEGER NWORDS.BUFSIZ.NMSGTX.NMSGRX.MTLSGN
      INTEGER DATA30
      INTEGER KL308.NWREC.KINC.KSM.NALREC.KDISP
C
      INTEGER I.J.IER.DMPREC(7).WARM
C
C
C
C     INITIALIZE TASKS
C
      CALL ITASK (COMCOM.10.1.IER)
      CALL ITASK (TRANSF.30.4.IER)
      CALL ITASK (ALMSGN.20.5.IER)
      CALL ITASK (GMAP.40.6.IER)
C
C     DEFINE IBI/3M 310 DMA CHANNEL TO NOVA 4/X
C
      CALL IMDEF (IER)
      TYPE "IMDEF IER =", IER
C
C     PROMPT FOR COLD OR WARM START
C
      WARM = 0
      WRITE (10, 25)
25    FORMAT ('DDO YOU WANT A COLD START? (Y/N) >', Z)
      READ  (11, 35) IANS
35    FORMAT (A2)
      IF (IANS .EQ. 'N ')
     *   WARM = 1
      IF (WARM .EQ. 1)  GO TO 100
C
C     COLD START => DELETE AND RE-CREATE OUTPUT FILES
C
      CALL DFILW ("WEATHER", IER)
      CALL DFILW ("DATA30", IER)
      CALL DFILW ("INCLOG", IER)
      CALL DFILW ("DATASM", IER)
      CALL DFILW ("ALMREC", IER)
      CALL DFILW ("QEWDYN", IER)
      CALL DFILW ("XMTTAB", IER)
      CALL DFILW ("DMPREC", IER)
      CALL DFILW ("RESDEV", IER)
      CALL DFILW ("TEMPFILE", IER)
C
      CALL CFILW("WEATHER".2.IER)
      CALL CFILW("DATA30".2.IER)
      CALL CFILW("INCLOG".2.IER)
      CALL CFILW("DATASM".2.IER)
      CALL CFILW("ALMREC".2.IER)
      CALL CFILW("QEWDYN".2.IER)
      CALL CFILW("XMTTAB".2.IER)
      CALL CFILW("DMPREC".2.IER)
      CALL CFILW("RESDEV".2.IER)
      CALL CFILW("TEMPFILE".2.IER)
      GO TO 999
C
C     WARM START => GET FILE POINTERS FROM FILE 'DMPREC'
C
100   CONTINUE
      CALL OPEN(2."DMPREC".1.IER.2*7)
      CALL READR(2.1.DMPREC(1).1.IER)
      KL308=DMPREC(1)
      NWREC=DMPREC(2)
      KINC=DMPREC(3)
      KSM=DMPREC(4)
      NALREC=DMPREC(5)
      KDISP=DMPREC(6)
      NWORDS=DMPREC(7)
      CALL CLOSE(2.IER)
C
      IF(NMSGTX.EQ.0)GO TO 999
      CALL OPEN (2."XMTTAB".1.IER.2*45)
      CALL READR(2,NWORDS,DATA30(1),1,IER)
      DO 150 I=1,NUFSIZ
        XMINUP(I)=DATA30(I+6)
150     CONTINUE
      CALL CLOSE(2,IER)
      GO TO 999
C
C     END OF ROUTINE
C
999   CONTINUE
      RETURN
      END
```

```
C
      COMPILER NOSTACK
      OVERLAY UINPEDIT
      SUBROUTINE INPEDIT(ICODE.INUM.ICHAR.NVAR.ICOUNT.ICHCNT.COM1.COM2)
C
C
      INTEGER COM1.COM2.ICODE(30).INUM(0130).ICHAR(60).IN(62).DIGIT
      INTEGER NUM(019)
      LOGICAL CHAR,CONTIG
C
      DATA NUM(0) /'0 '.'1 '.'2 '.'3 '.'4 '.'5 '.'6 '.'7 '.'8 '.'9 '/
C
C     THIS SUBROUTINE SEPARATES THE INTEGER INPUT FROM THE CHARACTER INPUT
C     ONLY CHARACTER AND NON-NEGATIVE INPUT IS ALLOWED
C
C     VARIABLES:
C     ----------
C
C             IN    -ARRAY HOLDING COMMAND STRING (ONE CHARACTER PER ELEMENT)
C             INUM  -ARRAY HOLDING INTEGER INPUT
C             ICHAR -ARRAY HOLDING CHARACTER INPUT
C                    AFTER EACH CHARACTER STRING  A BLANK IS PLACED
C             ICODE -ARRAY HOLDING ORDER IN WHICH CHARACTERS AND INTEGERS
C                    APPEAR IN COMMAND LINE
C                      1=INTEGER
C                      2=CHARACTER
C             ICOUNT-THE NUMBER OF INTEGERS
C             ICHCNT-THE NUMBER OF CHARACTERS
C             NVAR  -THE NUMBER OF VARIABLES
C             CONTIG-FLAG TO INDICATE IF DIGIT IS PART OF PREVIOUS VARIABLE
C             CHAR  -FLAG TO INDICATE A CHARACTER STRING
C
5     CONTIG=.FALSE.
      CHAR=.FALSE.
      DO 1 I=1.ICHCNT
1     ICHAR(I)=20040K
      DO 2 I=1.ICOUNT
2     INUM(I)=0
      DO 3 I=1.NVAR
3     ICODE(I)=0
      NVAR=0
      ICOUNT=0
      ICHCNT=0
7     WRITE(10.250)
      READ(11.300)IN
      IF(IN(61).EQ.'  '.AND.IN(62).EQ.'  ')GO TO 10
      TYPE "COMMAND LINE OVER 60 CHARACTERS LONG"
      GO TO 7
10    DO 200 L=1.60
      DO 15 N=0.9
      IF(NUM(N).EQ.IN(L))GO TO 100 ; NUMERIC
15    CONTINUE
      IF(CHAR)GO TO 30
      IF(.NOT.CONTIG)GO TO 20
      IF(IN(L).EQ.'  ')GO TO 45
17    TYPE "NO SPACE BETWEEN NUMERIC AND CHARACTER"
      GO TO 5
20    IF(IN(L).EQ.'  ')GO TO 200
      CONTIG=.TRUE.
      CHAR=.TRUE.
      NVAR=NVAR+1
      ICODE(NVAR)=2
30    ICHCNT=ICHCNT+1
      IF(IN(L).EQ.'  ')GO TO 40
      ICHAR(ICHCNT)=IN(L)
      GO TO 200
40    CHAR=.FALSE.
45    CONTIG=.FALSE.
      GO TO 200
100   IF(CHAR)GO TO 17
      IF(CONTIG)GO TO 110
      DIGIT=0
      NVAR=NVAR+1
      ICODE(NVAR)=1
      CONTIG=.TRUE.
      ICOUNT=ICOUNT+1
110   IF(INUM(ICOUNT).EQ.0.AND.N.EQ.0)GO TO 200
      DIGIT=DIGIT+1
      IF(DIGIT.LT.5)GO TO 150
      IF(IN(L+1).EQ.'  ')GO TO 140
120   TYPE "INTEGER OVERFLOW; NUMBER > 32767"
      GO TO 5
140   IF(INUM(ICOUNT).GT.3276)GO TO 120
      IF(INUM(ICOUNT).LT.3276)GO TO 150
      IF(N.GT.7)GO TO 120
```

```
150       INUM(ICOUNT)=INUM(ICOUNT)*10+N
200       CONTINUE
250       FORMAT(' >',Z)
300       FORMAT(62A1)
          IF(NVAR.NE.0)GOTO 320
          WRITE(10,310)COM1,COM2
310       FORMAT(/1X,2A2)
          GOTO 5
320       IF(ICHAR(1).NE.15400K)GOTO 400 ; ESC
          CALL &PKIL("@TTO",IER)
          GOTO 5
C
400       RETURN
          END
```

```
      OVERLAY OIOC2
      SUBROUTINE IOC2
C
CFUNCTION : PAINT THE EXTRA LANE ON WINSTON CHURCHILL BASED ON OCCUPANCY
C
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,RQEW,RWTEMP
      COMMON/EXTRA/WINSTON
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      LOGICAL WINSTON
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1,IDATA(39)
      INTEGER QEW(512),GCHN,PFILE,RQEW,RWTEMP
C
C     WINSTON CHURCHILL RD HAS AN EXTRA LANE CHECK IF DISPLAYING THIS MAP
C
      IF(WINSTON) RETURN  !TRUE IMPLIES LANE DATA HAS BEEN DISPLAYED
C
C     DETERMINE COLOUR TO PAINT EXTRA LANE
C
      IOEL=411
      IF(QEW(IOEL) .GT. 30) WRITE(GCHN,100)
      IF(QEW(IOEL) .GE. 20 .AND. QEW(IOEL) .LE. 30) WRITE(GCHN,400)
      IF(QEW(IOEL) .LT. 20) WRITE(GCHN,200)
      IF(QEW(IOEL) .LT. 0) WRITE(GCHN,800)
900   READ(PFILE,18) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 900
C
C     SET WINSTON=T AFTER PAINTING EXTRA LANE ONCE
C
      WINSTON=.TRUE.
      RETURN
18    FORMAT(39A2)
35    FORMAT(" ",39A2)
100   FORMAT(" ",":[COL C1 C0")
400   FORMAT(" ",":[COL C4 C0")
200   FORMAT(" ",":[COL C2 C0")
800   FORMAT(" ",":[COL C7 C0")
      END
      OVERLAY OIOCC
      SUBROUTINE IOCC
C
C
C FUNCTION: DETERMINE THE COLOUR TO PAINT THE INTERCHANGE LANES
C           BASED UPON THE 1 MINUTE MAINLANE DATA,ACCORDING TO OCC
C
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,RQEW,RWTEMP
      COMMON/OPTB/IODL,IOCL,IOPL
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,WO,RV,TXT,K,K1
      INTEGER NSTA,NCON,IS,IV,IO,WO,RV,TXT(6),K,K1,IDATA(39)
      INTEGER QEW(512),GCHN,PFILE,RQEW,RWTEMP
C
C     TEST OCCUPANCY ON PASSING LANE
C
      IF(QEW(IOPL) .GT. 30) WRITE(GCHN,100)
      IF(QEW(IOPL) .GE. 20 .AND. QEW(IOPL) .LE. 30) WRITE(GCHN,400)
      IF(QEW(IOPL) .LT. 20) WRITE(GCHN,200)
      IF(QEW(IOPL) .LT. 0) WRITE(GCHN,800)
C
C     PAINT PASSING LANE
C
900   READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 900
C
C     TEST OCCUPANCY ON CENTRE LANE
C
      IF(QEW(IOCL) .GT. 30) WRITE(GCHN,100)
      IF(QEW(IOCL) .GE. 20 .AND. QEW(IOCL) .LE. 30) WRITE(GCHN,400)
      IF(QEW(IOCL) .LT. 20) WRITE(GCHN,200)
      IF(QEW(IOCL) .LT. 0) WRITE(GCHN,800)
C
C     PAINT CENTRE LANE
C
1000  READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 1000
C
C     TEST OCCUPANCY ON DRIVING LANE
C
      IF(QEW(IODL) .GT. 30) WRITE(GCHN,100)
      IF(QEW(IODL) .GE. 20 .AND. QEW(IODL) .LE. 30)WRITE(GCHN,400)
      IF(QEW(IODL) .LT. 20) WRITE(GCHN,200)
      IF(QEW(IODL) .LT. 0) WRITE(GCHN,800)
C
C     PAINT DRIVING LANE
C
1100  READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1 ) GOTO 1100
      RETURN
10    FORMAT(39A2)
35    FORMAT(" ",39A2)
100   FORMAT(" ","|COL C1 C0")
400   FORMAT(" ","|COL C4 C0")
200   FORMAT(" ","|COL C2 C0")
800   FORMAT(" ","|COL C7 C0")
      END
      OVERLAY OLOOP
      SUBROUTINE LOOP
C
C     FUNCTION : TO FLASH POLYGONS AT ACCIDENT LOCATIONS (FIRST 4 LOC ONLY)
C
      COMMON/SCREEN/GCHN,PFILE,RQEW,RWTEMP
      COMMON/MACC/KCNT
      INTEGER IDATA(39)
C
      INTEGER GCHN,PFILE,RQEW,RWTEMP
C
C     LOOP ON TEMPFILE N TIMES TO FLASH THE ACCIDENT POLYGONS BTW C7 & C1
C
      IF(KCNT .EQ. 1) N=5
      IF(KCNT .EQ. 2) N=4
      IF(KCNT .EQ. 3) N=2
C
C     SET COLOUR TO BLACK (C7)
C
      DO 800 J=1,N
      WRITE(GCHN,1005)
      REWIND RWTEMP
50    READ(RWTEMP,100,END=700) (IDATA(I),I=1,39)
      WRITE(GCHN,200)       (IDATA(I),I=1,39)
      GOTO 50
100   FORMAT(39A2)
200   FORMAT(" ",39A2)
C
C     CHANGE COLOUR TO RED (C1)
C
700   WRITE(GCHN,1007)
      REWIND RWTEMP
750   READ(RWTEMP,100,END=800) (IDATA(I),I=1,39)
      WRITE(GCHN,200)       (IDATA(I),I=1,39)
      GOTO 750
800   CONTINUE
      CALL CLOSE(RWTEMP,IER)
      CALL DELETE("TEMPFILE")
      RETURN
1005  FORMAT(" ","[COL C7 C0")
1007  FORMAT(" ","[COL C1 C0")
      END
      SUBROUTINE MISCOM
C
C     A ONE-MINUTE PERIODIC COMMUNICATION PROGRAM,
C       TO EXCHANGE SURVEILLANCE AND CONTROL DATA WITH
C       THE MISSISSAUGA TRAFFIC CONTROL COMPUTER SYSTEM.
C
C
      RETURN
      END
```

```
C
      SUBROUTINE MODM1
C
      COMMON/COMM/DETECT(545),NDETNC
      COMMON/MD/MODEMT(299),MODEM(148),MODEM1(73),NUMMOD
      COMMON/PROT/CONMSG(2),REPLY(2),PROTOCOL,MSGID,KTIME,PSPRT
      COMMON/LIMITS/MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      COMMON/MDTBL/NERROR(73),MDMDET(73),XTH(180)
      COMMON/DINDX/XNEFTH(180),XNEF(113)
C
      INTEGER MODEMH(148)
      INTEGER DETECT, NDETNC
      INTEGER MODEMT,MODEM,MODEM1,NUMMOD
      INTEGER CONMSG,REPLY,PROTOCOL,MSGID,KTIME,PSPRT
      INTEGER MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      INTEGER NERROR,MDMDET,XTH
      INTEGER XNEFTH,XNEF
      INTEGER DIFF,J,ID1,ID2,NO,LIMIT
C
      EQUIVALENCE (MODEMH(1),MODEMT(152))
C
C
C
      DO 100 I=1,NUMMOD
      IF(I.EQ.3)GOTO 100; STGN MODEM
      IF(I.LE.8)GOTO 100; MODEMS 1 TO 8 NOT USED
      IF(I.EQ.12.OR.I.EQ.14.OR.I.EQ.20.OR.I.EQ.24)GOTO 100;NOT USED
      IF(I.EQ.40.OR.I.EQ.46.OR.I.EQ.47.OR.I.EQ.48)GOTO 100;NOT USED
      IF(I.EQ.56.OR.(I.GE.59.AND.I.LE.64).OR.I.EQ.68)GOTO 100;NOT USED
      DIFF=MAXOCC-DETECT(470+I)
      IF(DIFF.LT.0)DIFF=-DIFF
      IF(DIFF.LE.2)DIFF=0 ;ALLOW DEVIATION OF 2 SAMPLES
      MODEMT(I)=MODEMT(I)+DIFF ;ACCUMULATE ERROR COUNTS
      NO=MOD(MDMDET(I),10) ;GET NO OF DET'S AT THAT ADDRESS
      IF(NO.LE.0)GOTO 100 ;NO OF DET'S AT THAT ADDRESS
      ID1=MDMDET(I)/10 ;GET START LOC OF FIRST DETECTOR
      ID2=ID1+NO-1 ;GET ENDING LOCATION
      IF(NERROR(I).GE.5)GOTO 120 ;LOOK FOR MODEM RECOVERY
      IF(DETECT(470+I).GT.0)GOTO 140 ;MODEM IS RESPONDING
      NERROR(I)=NERROR(I)+1 ;INC ERROR COUNTS
      IF(NERROR(I).LT.5)GOTO 100 ;NOT PERMANENT ERROR YET
      DO 110 J=ID1,ID2
110   XNEFTH(XTH(J))=MOD(XNEFTH(XTH(J)),1000)+1000
150   NERROR(I)=5
      GOTO 100
120   LIMIT=MAXOCC-2
      IF(DETECT(470+I).LT.LIMIT)GOTO 150 ;MODEM STILL IN ERR STATE
      NERROR(I)=NERROR(I)+1 ;INC RECOVERY COUNTER
      IF(NERROR(I).LT.10)GOTO 100 ;NOT RECOVERED YET
      DO 160 J=ID1,ID2
160   XNEFTH(XTH(J))=MOD(XNEFTH(XTH(J)),1000)
140   NERROR(I)=0 ;RESET NO RESPONSE ERROR COUNTER
100   CONTINUE
C
      KTIME=KTIME+1 ;COUNTER TO IDENTIFY 10-MIN TIME PERIOD
      IF(KTIME.LT.PSPRT) GO TO 999
      DO 200 I=1,NUMMOD
      MODEMH(I)=MODEM1(I)
200   MODEM1(I)=0
C
999   RETURN
      END
```

```
                     --------------
C
      SUBROUTINE NXFEW
C
      DIMENSION SDVOL(46),SDOCC(46)
      DIMENSION DNVOL(67),UPVOL(67),DNOCC(67),UPOCC(67),LNSPD(67)
      DIMENSION LVCNT(67)
C
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/LIMITS/MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      COMMON/DIS/VOL1S(46),OCC1S(46),DVOL1S(67),UVOL1S(67),DOCC1S(67),
     *UOCC1S(67),VSPEED(67),LVC1S(67),LTCOND(10)
      COMMON/ET/STIM1(46),DTIM1(67),DTIM2(67),UPTIM(67)
      COMMON/ALM1/APARM(6)
      COMMON/DINDX/XREFTB(180),XREF(113)
      COMMON/D30/DATA30(718)
      COMMON/CTRL4/CHEK1(10),CHEK2(10),LIGHTS(10)
C
      INTEGER SDVOL,SDOCC
      INTEGER DNVOL,UPVOL,DNOCC,UPOCC,LNSPD,LVCNT
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      INTEGER VOL1S,OCC1S,DVOL1S,UVOL1S,DOCC1S
      INTEGER UOCC1S,VSPEED,LVC1S,LTCOND
      INTEGER STIM1,DTIM1,DTIM2,UPTIM
      INTEGER APARM
      INTEGER XREFTB,XREF
      INTEGER DATA30
      INTEGER CHEK1,CHEK2,LIGHTS
      INTEGER IDX
C
      EQUIVALENCE (SDVOL(1),DATA30(49)),(SDOCC(1),DATA30(95)),
     *(DNVOL(1),DATA30(141)),(UPVOL(1),DATA30(208)),
     *(DNOCC(1),DATA30(275)),(UPOCC(1),DATA30(342)),
     *(LNSPD(1),DATA30(409)),(LVCNT(1),DATA30(476))
C
      DO 100 I=1,DTOT
      IF(DTIM1(I).LT.APARM(2).AND.DTIM2(I).LT.APARM(2).AND.
     *UPTIM(I).LT.APARM(2))GOTO 100 !NO VEH STOPPED OVER DETECTOR
      VSPEED(I)=VSPEED(I)+S1/10 !BECAUSE VEH STOPPED OVER DET
      DTIM1(I)=0
      DTIM2(I)=0
      UPTIM(I)=0
100   CONTINUE
      DO 110 I=1,STOT
      IF(STIM1(I).GE.APARM(2))STIM1(I)=0 !REINIT TO AVOID OVERFLOW
110   CONTINUE
      DO 200 I=1,STOT
      IDX=XREF(I)
      SDVOL(IDX)=VOL1S(I)
      SDOCC(IDX)=OCC1S(I)
      VOL1S(I)=0 !REINIT FOR 30 SECS
      OCC1S(I)=0
200   CONTINUE
      DO 250 I=1,DTOT
      IDX=XREF(STOT+I)
      DNVOL(IDX)=DVOL1S(I)
      UPVOL(IDX)=UVOL1S(I)
      DNOCC(IDX)=DOCC1S(I)
      UPOCC(IDX)=UOCC1S(I)
      LNSPD(IDX)=(VSPEED(I)+5)/10 !M/S
      LVCNT(IDX)=LVC1S(I)
      DVOL1S(I)=0 !REINIT FOR NEXT 30 SECS
      UVOL1S(I)=0
      DOCC1S(I)=0
      UOCC1S(I)=0
      LVC1S(I)=0
250   CONTINUE
C
      DO 300 I=1,NRAMPS
      LIGHTS(I)=LTCOND(I) !30 SEC LIGHT STATUS
      LTCOND(I)=0 !REINIT FOR NEXT 30 SECS
300   CONTINUE
C
      RETURN
      END
```

```
C
      OVERLAY OPCBOX
      SUBROUTINE PCBOX
C
C     ABSTRACT:
C
C       THIS ROUTINE DRAWS THE HIGHWAY 2 & HIGHWAY 5 PCS DATA
C     BOXES ON THE MAIN QEW DISPLAY.
C
C     SPECIFICATION STATEMENTS:
C
C
C     COMMON STATEMENTS:
C
      COMMON /QEWDATA/ QEW
      COMMON /SCREEN/ GCHN, PFILE, RQEW, RWTEMP
C
C     DATA-TYPE STATEMENTS:
C
C
C                          INPUT VARIABLES
C
      INTEGER GCHN                 !GRAPHIC TERMINAL CHANNEL
      INTEGER PFILE                !POLYGON FILE CHANNEL
      INTEGER QEW(512)             !RECORD IN 'QEWDYN' FILE
      INTEGER RQEW                 !READ 'QEWDYN' CHANNEL
      INTEGER RWTEMP               !READ/WRITE 'TEMPFILE' CHANNEL
C
C
C                          INTERNAL VARIABLES
C
      INTEGER I, J                 !INDICES
      INTEGER IDATA(39)            !RECORD IN POLYGON FILE
      INTEGER IO, IS, IV           !INDEX OCC., SPD., VOL.
      INTEGER NBOX                 !NUMBER OF BOXES TO DRAW
      INTEGER OCC                  !OCCUPANCY
      INTEGER SPD                  !SPEED
      INTEGER VOL                  !VOLUME
C
C
C     FORMAT STATEMENTS:
C
105   FORMAT (2I4)
115   FORMAT (39A2)
C
205   FORMAT (' ', '[COL C5 C0')
215   FORMAT (' ', 39A2)
225   FORMAT (' ', '[AVEC 56,0 0,56 -56,0 0,-56 56,0')
235   FORMAT (' ', '[AVEC -56,35[STR /', I3, ' SPD/')
245   FORMAT (' ', '[AVEC 0,-14[STR /', I3, ' VOL/')
255   FORMAT (' ', '[AVEC 0,-14[STR /', I3, ' OCC/')
C
C
C     EXECUTABLE STATEMENTS:
C
C
C                               SET COLOUR FOR BOXES
C
      WRITE (GCHN, 205)
C
C                               GET NUMBER OF BOXES AND SPEED INDEX
C
      READ (PFILE, 105) NBOX, IS
C
C                               SET VOLUME AND OCCUPANCY INDICES
C
      IV = IS + 1
      IO = IV + 1
C
C                               BEGIN LOOP ON BOXES
C
      IF (NBOX .LT. 1) GO TO 9999
      DO 1900  I = 1, NBOX
C
C                                   DRAW BOX OUTLINE
C
         READ (PFILE, 115) (IDATA(J), J=1,39)
         WRITE (GCHN, 215) (IDATA(J), J=1,39)
         WRITE (GCHN, 225)
C
C                                   OUTPUT SPEED
C
         SPD = (QEW(IS) + QEW(IS+3)) / 2
         WRITE (GCHN, 235) SPD
      IS = IS + 2*3
C
C                                OUTPUT VOLUME
C
      VOL = WEN(IV) + QEN(IV+4)
      WRITE (GCHN, 245) VOL
      IV = IV + 2*3
C
C                                OUTPUT OCCUPANCY
C
      OCC = (QEN(IO) + QEN(IU+3)) / 2
      WRITE (GCHN, 255) OCC
      IO = IO + 2*3
C
C                               END OF LOOP
C
1900    CONTINUE
      GO TO 9999
C
C                              END OF ROUTINE
C
9999  CONTINUE
      RETURN
      END
      SUBROUTINE PCSPOL
C
C
      RETURN
      END
      OVERLAY OPENCHK
      SUBROUTINE PENCHK
      DIMENSION INCLOG(25)
      DIMENSION CSOCC(22),PSOCC(22),SOCCIM(22),SVOLIM(22)
      COMMON/INCA/INCTV(84),INCREC(21),INCFLAG(21),INCSTAT(21)
      COMMON/INCB/OCCOBI(21),OCCUBI(21),INCNT(21),INCONT,INCEX
      COMMON/INCC/THHMQL
      COMMON/INCE/CHCK(21),CMNI1(21),CMNJ(21),TEST3
      COMMON/[illegible]/HMS(3),MDY(3)
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/SECTL/DIST(20)
      COMMON/TTS/TT(20)
      COMMON/D30/DATA30(718)
      COMMON/NREC/KL308,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER INCLOG
      INTEGER CSOCC,PSOCC,SOCCIM,SVOLIM
      INTEGER INCTV,INCREC,INCFLAG,INCSTAT
      INTEGER OCCOBI,OCCUBI,INCNT,INCONT,INCEX
      INTEGER THHMQL
      INTEGER HMS,MDY
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER DIST
      INTEGER TT
      INTEGER DATA30
      INTEGER KL308,NWREC,KINC,KSM,NALREC,KDISP
      INTEGER NSTNM1,I1,J,K,KK,IK
      INTEGER MQL,IJ,IJ1,ITT,ISUM
      INTEGER CHCK,CMNJ,CMNI1,TEST3
      EQUIVALENCE (CSOCC(1),DATA30(543)),(PSOCC(1),DATA30(565)),
     *(SOCCIM(1),DATA30(587)),(SVOLIM(1),DATA30(609))
C
C
      NSTNM1=NSTNS-1
      DO 100 I=1,NSTNM1
      I1=CMNI1(I)
      J =CMNJ (I)
C
C 1-MIN PERSISTENCE CHECK
C INC DETECTED FOR LESS THAN 1 MIN IS REJECTED
C
      IF(CHCK(I).EQ.0)GOTO 100 ;        IF NO INCIDENT WAS DETECTED, GO TO NEXT STATION
      IF(INCFLAG(I).EQ.0.AND.INCSTAT(I).EQ.0)GOTO 30 ;NO TRUE INC
      IF(INCFLAG(I).EQ.INCSTAT(I))GOTO 100 ;TRUE INC STILL ON
      IF(INCFLAG(I).EQ.0)GOTO 110 ;TRUE INC CLEARED
      IF(TEST3.LT.INCTV(4*I-1)) GOTO 30
      INCNT(I)=INCNT(I)+1 ;TIME COUNTER FOR 1-MIN CHECK
      IF(INCNT(I).NE.3)GOTO 100 ;NOT 1-MIN YET
      INCSTAT(I)=1 ;INC CONFIRMED BY 1-MIN CHECK
      [illegible] ;TO ALARM OPERATOR
      INCLOG(1)=MDY(3) ;REC DATE
      INCLOG(2)=MDY(1)
      INCLOG(3)=MDY(2)
      INCLOG(4)=I ;REC STN ID
      INCLOG(5)=0 ;LANE
      INCLOG(6)=HMS(1)
      INCLOG(7)=HMS(2)-1
      INCLOG(8)=HMS(3)
      IF(INCLOG(7).GE.0)GOTO 90
      INCLOG(6)=HMS(1)-1
      INCLOG(7)=59
C CALCULATE MAX QUEUE LENGTH
90    MQL=0 ;QUEUE LENGTH OUT OF CONTROL SECTION
      DO 92 IJ=1,I1
      IJ1=I1-IJ+1 ;STN UPSTR OF INC LOCATIONS TO BE SEARCHED
      IF(IJ1.EQ.1)GOTO 92 [illegible]
      IF(CSOCC(IJ1).GE.THHMQL.OR.CSOCC(IJ1).LT.0)GOTO 92
      MQL=IJ1 ;MAINLINE QUEUE LENGTH FOUND
      GOTO 94
92    CONTINUE
94    INCLOG(13)=MQL   ;MAX QUEUE LENGTH
C
      INCLOG(14)=SVOLIM(J) ;DOWNSTREAM VOL
      INCLOG(15)=CAP(J)-INCLOG(14)*60 ;MAX CAP REDUCTION
C CALCULATE TOTAL SECTION TRAVEL TIMES
      ITT=-1
      ISUM=0
      DO 52 IJ=1,20
      IF(TT(IJ).LT.0)GOTO 54
      ISUM=ISUM+TT(IJ)
52    CONTINUE
      ITT=ISUM
54    INCLOG(16)=ITT ;TT
C
      DO 120 IK=17,25
120   INCLOG(IK)=0
```

```
      INCREC(I)=KINC ;SAVE REC NO FOR CLEARANCE
      KINC=KINC+1
      CALL OPEN(3,"INCLOG",2,IER,2*25)
      GOTO 80
110   INCSTAT(I)=0 ;TRUE INC CLEARED
      CALL OPEN(3,"INCLOG",2,IER,2*25)
      CALL READR(3,INCREC(I),INCLOG,1,IER)
      INCLOG(9)=HMS(1) ;END TIME OF INC
      INCLOG(10)=HMS(2)
      INCLOG(11)=HMS(3)
      INCLOG(12)=((HMS(1)-INCLOG(6))*60+HMS(2)-INCLOG(7))*60+HMS(3)-
     *   INCLOG(8) ;INC DURATION
80    CALL WRITR(3,INCREC(I),INCLOG,1,IER)
      CALL CLOSE(3,IER)
      IF(INCSTAT(I).EQ.0)INCREC(I)=0
30    INCNT(I)=0
100   CONTINUE
      RETURN
      END
      OVERLAY QOCC0
      SUBROUTINE QOCC
C
C FUNCTION : COLOUR MAINLINE POLYGONS ACCORDING TO OCCUPANCY
C
C
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,NQEW,NWTEMP
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1
      INTEGER GCHN,PFILE,IDATA(39)
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER QEW(512),NQEW,NWTEMP
C
C***  COLOUR BASED ON OCCUPANCY***
C
C     I1 = INITIAL WORD FOR OCCUPANCY OR SPEED AT STATION I
C     I1-5 REPRESENT THE WORD LOCATION FOR OCC AND SPEED AT THEIR
C             RESPECTIVE STATIONS. THERE ARE PRESENTLY 22 STATIONS
C          ALONG THE MAINLINE.
C
      I1=170
C
C     SET COLOUR BY LOOPING ON THE TOTAL # OF MAINLINE STATIONS
C
      DO 100 I=1,NSTNS
      I1=I1-5
      IF(QEW(I1) .GT. 30) WRITE(GCHN,300)
      IF(QEW(I1) .GE. 20 .AND. QEW(I1) .LE. 30) WRITE(GCHN,350)
      IF(QEW(I1) .LT. 20) WRITE(GCHN,375)
      IF(QEW(I1) .LT. 0) WRITE(GCHN,400)
400   FORMAT(" ","[COL C7 C0")
300   FORMAT(" ","[COL C1 C0")
350   FORMAT(" ","[COL C4 C0")
375   FORMAT(" ","[COL C2 C0")
C
C
C      TO EXTRACT THE POLYGON TO BE COLOURED ON THE Q DISPLAY
C
C
390   READ(PFILE,10) (IDATA(J),J=1,39)
      WRITE(GCHN,35) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 390
100   CONTINUE
10    FORMAT(39A2)
35    FORMAT(" ",39A2)
      RETURN
      END
      OVERLAY USSPD
      SUBROUTINE USPD
C
C FUNCTION      :READS TOTAL NUMBER OF STATIONS & METERED RAMPS FOR THE MAIN DISPLAY
C               -READS MAP FILE AND DISPLAYS HEADINGS
C               -DISPLAYS TIME & DATE FROM QEWDYN
C               -COLOURS MAINLINE POLYGONS ACCORDING TO SPEED
C
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,HQEW,RWTEMP
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/OCCRM/OCC,RMETER
      COMMON/WORDS/NSTA,NCON,IR,IV,IO,RO,RV,TXT,K,K1
C
      INTEGER NSTA,NCON,IR,IV,IO,RO,RV,TXT(6),K,K1
      INTEGER CHN1,CHN2,CHN3,CHN8,CHN5,CHN12,CHN13,CHN14,CHN15
      INTEGER GCHN,PFILE,IDATA(39)
      INTEGER QEW(512)
      INTEGER HQEW,RWTEMP
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
C
      LOGICAL  RMETER(10),OCC
C
C
C     READ AND PLACE STRING HEADING
C
      READ(PFILE,18) (IDATA(I),I=1,39)
18    FORMAT(39A2)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
35    FORMAT(" ",39A2)
      IF(OCC) WRITE(GCHN,5500)
5500  FORMAT(" ",*IRVEC 120,0ISTR/(OCC)/*")
      IF(.NOT. OCC) WRITE(GCHN,6000)
6000  FORMAT(" ",*IRVEC 120,0ISTR/(SPD)/*")
C
C     DISPLAY CURRENT TIME AND DATE FROM QEWDYN
C
      WRITE(GCHN,30) (QEW(I),I=1,3)
      WRITE(GCHN,37) (QEW(I),I=4,6)
30    FORMAT(" ",*IRVEC 100,0ISTR/DATE:*,3(1X,I2),*/*")
37    FORMAT(" ",*IRVEC 240,0ISTR/TIME:*,3(1X,I2),*/*")
C
C     DETERMINE COLOURING OPTION
C
      IF(OCC) RETURN         !            RETURN IF COLOR BY OCCUPANCY
C
C *** COLOUR BASED ON SPEED  ***
C
      I1=160
      DO 80 I=1,NSTNS
      I1=I1-5
      IF(QEW(I1) .GT. 100) WRITE(GCHN,3000)
      IF(QEW(I1) .GE. 80 .AND. QEW(I1) .LE. 100) WRITE(GCHN,375)
      IF(QEW(I1) .GE. 60 .AND. QEW(I1) .LT. 80) WRITE(GCHN,350)
      IF(QEW(I1) .GE. 40 .AND. QEW(I1) .LT. 60) WRITE(GCHN,400)
      IF(QEW(I1) .LT. 40) WRITE(GCHN,300)
      IF(QEW(I1) .LT. 0) WRITE(GCHN,380)
C
3000  FORMAT(" ",*ICOL C3 C0*")
300   FORMAT(" ",*ICOL C1 C0*")
350   FORMAT(" ",*ICOL C6 C0*")
375   FORMAT(" ",*ICOL C2 C0*")
380   FORMAT(" ",*ICOL C7 C0*")
400   FORMAT(" ",*ICOL C5 C0*")
C
500   READ(PFILE,18) (IDATA(J),J=1,39)
      WRITE(GCHN,35) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 500
80    CONTINUE
C
      RETURN
      END
      OVERLAY QUQOCC
      SUBROUTINE QUOCC
      DIMENSION LQOCC(13)
      DIMENSION QUETAB(256),MQOCC(10).QL(2).CYCLES(50)
      COMMON/QQ1/QOVR(10),QIND(10)
      COMMON/QQ2/QWAIT(10).QLNTH(10)
      COMMON/QQ3/KOCNT(10).KSTEP(10).KTLMT(10).KTHR
      COMMON/QQ4/QOCCLB(10).QOCCUB(10)
      COMMON/CTRL1/CONTRL.MAXCYC.METER(10).MODCYC(10)
      COMMON/CTRL2/CYCOVR(10).RAMCYC(10)
      COMMON/GEOM/STOT.DTOT.NRAMPS.NSTNB.CAP(22)
      COMMON/D30/DATA30(718)
      COMMON/QUET/NPO,NOD.NEO.NQD
      COMMON/QOC/QOCC(13)
C
      INTEGER LQOCC
      INTEGER QUETAB,MQOCC,QL,CYCLES
      INTEGER QOVR.QIND
      INTEGER QWAIT.QLNTH
      INTEGER KOCNT.KSTEP.KTLMT.KTHR
      INTEGER QOCCLB.QOCCUB
      INTEGER CONTRL.MAXCYC.METER.MODCYC
      INTEGER CYCOVR.RAMCYC
      INTEGER STOT.DTOT.NRAMPS.NSTNB.CAP
      INTEGER DATA30
      INTEGER NPD.NOD.NED.NQD
      INTEGER QOCC
      INTEGER IDX.K.I.N1.N2.I1.I2.IJ.J.ID.ID1
      EQUIVALENCE (LQOCC(1).DATA30(126))
C
      CALL OPEN(5,"QUETAB".2.IER.2*64)
      CALL READR(5,1.QUETAB.4.IER)
      CALL CLOSE(5,IER)
      CALL OPEN(5,"CYCLES".2.IER.2*50)
      CALL READR(5,1.CYCLES.1.IER)
      CALL CLOSE(5.IER)
C
C QUEUE LENGTH COMPUTATION USING TABLE LOOK-UP FROM "QUETAB"
C WHICH CONTAINS OBSERVED Q-OCC VS Q-LNTH DATA
C
      IF(CONTRL.EQ.0)RETURN
      IDX=NQD*9 ;THRES VALUES(OCC).9 PER Q-DETECTOR
      K=0
      DO 100 I=1,NRAMPS
      K=K+1 ;INDEX TO LOOK UP QOCC FOR THIS RAMP
      IF((CONTRL.EQ.1.OR.CONTRL.EQ.3).AND.METER(I).EQ.0)GOTO 170;NO METER
      IF(RAMCYC(I).EQ.6)GOTO 170
      IF(I.EQ.6.OR.I.EQ.7.OR.I.EQ.10)GOTO 20 ;RAMPS WITH 2 Q-DETECTORS
      IF(QOCC(K).GE.0)GOTO 60 ;RAMPS WITH 1 Q-DETECTOR
      GOTO 50
C RAMPS WITH 2 Q-DETECTORS
20    IF(QOCC(K).GE.0.OR.QOCC(K+1).GE.0)GOTO 40
50    QWAIT(I)=-1 ;SET TO -1 FOR HARDWARE FAIL.
      QLNTH(I)=-1
      QIND(I)=0
      GOTO 110
40    MQOCC(I)=MAX0(QOCC(K),QOCC(K+1))
      N1=(K-1)*9+1 ;FIRST Q-DETECTOR OCC INDEX
      N2=N1+9 ;2ND Q-DETECTOR INDEX
      J=-1  ;2 Q-DETECTORS ON THIS RAMP
      DO 48 I1=N1,N2,9
      I2=I1+8
      J=J+1
      DO 46 IJ=I1,I2
      IF(QOCC(K+J).LT.QUETAB(IJ))GOTO 44
      IF(IJ.NE.I2)GOTO 46
      ID=IDX+IJ+J+K ;INDEX TO LOCATE CORR Q-LNTH
      GOTO 42
44    ID=IDX+IJ+K+J-1 ;INDEX TO LOCATE CORR Q-LNTH
42    QL(J+1)=QUETAB(ID) ;QLNTH FOR Q-DETECTORS
      GOTO 48
46    CONTINUE
48    CONTINUE
      QLNTH(I)=MAX0(QL(1).QL(2)) ;Q-LNTH FOR RAMPS 7 & 10
      IF(I.EQ.6)QLNTH(I)=QL(1)+QL(2) ;Q- LNTH FOR R6
      GOTO 70
C
C RAMPS WITH 1 Q-DETECTOR
C
60    MQOCC(I)=QOCC(K)
      I1=(K-1)*9+1
      I2=I1+8
      DO 62 IJ=I1.I2
      IF(QOCC(K).LT.QUETAB(IJ))GOTO 64
      IF(IJ.NE.I2)GOTO 62
      ID=IDX+IJ+K
      GOTO 63
64    ID=IDX+IJ+K-1 ;INDEX TO LOCATE CORR Q-LNTH
63    QLNTH(I)=QUETAB(ID)
      GOTO 70
62    CONTINUE
C
C
70    IF(QLNTH(I).GT.100)QLNTH(I)=100
C
C Q-OVERRIDE CONTROL BASED ON Q-OCC
C
      IF(QOVR(I).EQ.1.OR.CYCOVR(I).NE.0)GOTO 170 ;QOVR DISABLED OR MAN.
      IF(QIND(I).EQ.1.AND.MQOCC(I).GE.QOCCLB(I))GOTO 150;QOVR STILL ON
      KQCNT(I)=0 ;RESET QOVR STEP-DOWN TIME COUNTER
      KTLMT(I)=0 ;RESET TIME LIMIT COUNTER
      IF(MQOCC(I).LT.QOCCUB(I))GOTO 170 ;<UPPER Q. THRESHOLD;NO QOVR
      QIND(I)=1 ;TO IND. QOVR ON
      MQOCYC(I)=4 ;QOVR MODE
150   IF(RAMCYC(I).LE.1)GOTO 180
      IF(KQCNT(I).GE.KSTEP(I))KQCNT(I)=0 ;TIME TO REDUCE 1 CYCLE
      IF(KQCNT(I).EQ.0)RAMCYC(I)=RAMCYC(I)-1 ;YES
      KQCNT(I)=KQCNT(I)+1 ;INCRE COUNT FOR CYC LNTH STEPDOWN
180   KTLMT(I)=KTLMT(I)+1 ;INCRE COUNT FOR P.G.
      IF(KTLMT(I).GE.KTMX)RAMCYC(I)=0 ;P. GREEN
      GOTO 160
170   QIND(I)=0
160   ID1=(I-1)*MAXCYC+RAMCYC(I)
      QWAIT(I)=QLNTH(I)*20 ;ASSUME 2 SECS FOR P GREEN
      IF(RAMCYC(I).GT.0.AND.RAMCYC(I).LT.6)QWAIT(I)=QLNTH(I)*CYCLES(ID1)
110   IF(I.EQ.6.OR.I.EQ.7)K=K+1
100   CONTINUE
      RETURN
      END
```

```
C
      COMPILER NOSTACK
      SUBROUTINE RAMCON
C
C THIS ROUTINE ACCESSES 'RAMCYC' AND SETS UP CONTROL COMMANDS IN
C 'COMDOP' TO BE TRANSMITTED TO 2 DIFFERENT TYPES OF CONTROLLERS
C (CR710 AND C170). IT ALSO CHECKS TRANSMISSION ERRORS
C
      COMMON/COMM/DETECT(5*9),NDETRC
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/CTRL3/CNTADR(10),CSTAT(10),LTSTAT(10)
      COMMON/CTRL4/CHEK1(10),CHEK2(10),LIGHTS(10)
      COMMON/CTRL6/COMDOP(8),CNTON(10),NBRAMP
      COMMON/CTRL7/ERRCNT(10),OFRED
      COMMON/CTRL9/MCNT(10)
      INTEGER DETECT, NDETRC
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER CYCOVR,RAMCYC
      INTEGER CNTADR,CSTAT,LTSTAT
      INTEGER CHEK1,CHEK2,LIGHTS
      INTEGER COMDOP,CNTON,NBRAMP
      INTEGER ERRCNT,OFRED
      INTEGER XMT170(5),RCV170(5),ERRBS(5)
      INTEGER MCNT
      INTEGER IDX,K,LMT
C
      DATA ERRBS(1) /5*0/
      DATA XMT170(1) /5*0/, RCV170(1) /5*0/
C
C
C
C LOOK FOR MANUAL OVERRIDE
C
      DO 260 I=1,NRAMPS
      IF(METER(I).EQ.0)MODCYC(I)=0 ;RAMP NOT METERED
      IF(CYCOVR(I).EQ.8)GOTO 260 ;NO OVERRIDE
      IF(CYCOVR(I).EQ.9)GOTO 270 ;OVERRIDE TO DROP OFFLINE
      METER(I)=1 ;THIS RAMP IS METERED NOW
      RAMCYC(I)=CYCOVR(I) ;UPDATE RAMCYC ACCORDINGLY
      MODCYC(I)=5 ;MANUAL CONTROL MODE
      GOTO 260
270   METER(I)=0
      RAMCYC(I)=0 ;TO DROP OFFLINE
260   CONTINUE
C
C DETERMINE WHETHER THIS TIME SLICE IS FOR TRANSMITTING COMMANDS TO
C C170 N/B OR S/B RAMPS
C
      NBRAMP=NBRAMP+1
      IF(NBRAMP.EQ.1)GOTO 100 ;TIME FOR C170 N/B RAMPS
      NBRAMP=0 ;TIME FOR C170 S/B RAMPS
C
C SET UP CONTROL COMMANDS FOR CR710; H.O.BYTE FOR COMMAND,L.O.BYTE
C FOR ADDRESS
C
100   DO 300 I=1,NRAMPS
      IF(I.LE.2.OR.I.GE.8)GOTO 300 ;C170
      IDX=MCNT(I)
      COMDOP(IDX)=(RAMCYC(I)+RAMCYC(I)+49)*256+CNTADR(I) ;HI-0011***1,LOW-ADDR
      IF(METER(I).EQ.0)COMDOP(IDX)=0 ;DROP CNTLER OFFLINE
300   CONTINUE
C
C SETS UP CONTROL COMMANDS FOR C170
C
      DO 400 I=1,NRAMPS
      IF(I.GE.3.AND.I.LE.7)GOTO 400 ;CR710
      IDX=MCNT(I)
      IF(I.EQ.10)GOTO 450 ;C170 WITH 1 RAMP ONLY
      IF(NBRAMP.EQ.1)GOTO 420 ;TIME FOR C170 N/B RAMPS
      IF(I.EQ.1.OR.I.EQ.8)GOTO 400 ;N/B RAMPS
      GOTO 450
420   IF(I.EQ.2.OR.I.EQ.9)GOTO 400 ;S/B RAMPS
450   COMDOP(IDX)=(RAMCYC(I)+16)*256+CNTADR(I) ;HI-0001****,LOW-ADDR
      IF(NBRAMP.EQ.1.OR.I.EQ.10)CALL ISET(COMDOP(IDX),15) ;HI-1001****
      IF(I.EQ.10)GOTO 470
      IF(NBRAMP.EQ.0)CALL ISET(COMDOP(IDX),14) ;HI-0101****
470   IF(CYCOVR(I).NE.9)GOTO 490
      CALL ICLR(COMDOP(IDX),12) ;HOLD-ON LINE BIT OFF
      IF(CNTON(I).EQ.0)COMDOP(IDX)=0 ;DROP OFFLINE AFTER SENDING HOL BIT 0
      CNTON(I)=0
      GOTO 400
490   CNTON(I)=1
```

```
600     CONTINUE
C
C STORE COMMANDS XMTED TO C170
C
        K=1
        IF(NBRAMP.EQ.0)K=2
        XMT170(K)=COMDOP(8)
        K=K+2
        XMT170(K)=COMDOP(6)
        XMT170(5)=COMDOP(7)
C
C STORE RETURN COMMANDS FOR C170
C
        K=1
        IF(ITEST(DETECT(8),6))K=2 ;8/9 RAMP
        RCV170(K)=DETECT(8)
        K=3
        IF(ITEST(DETECT(6),6))K=4 ;8/9 RAMP
        RCV170(K)=DETECT(6)
        RCV170(5)=DETECT(7)
C
C CHECK FOR TRANSMISSION ERRORS-MAKE SURE XMTED=RECEIVED
C
        DO 700 I=1,NRAMPS
        IF(I.LE.2.OR.I.GE.8)GOTO 700 ;C170 CONTROLLERS
        IDX=NCNT(I)
        IF(CSTAT(I).EQ.1)GOTO 690 ;ERROR ALREADY NOTED
        IF(METER(I).EQ.0)GOTO 690 ;RAMP NOT METERED
        CHEK1(I)=(DETECT(IDX).AND.62)*256 ;GET BITS 1-5 & MOVE LEFT 8 BITS
        CHEK2(I)=COMDOP(IDX).AND.15872 ;RETAIN BITS 9-13
        IF(CHEK1(I).EQ.CHEK2(I))GOTO 690
        ERRCNT(I)=ERRCNT(I)+1 ;INC ERROR COUNTS
        IF(ERRCNT(I).LE.OFRES)GOTO 695 ;NO OF ERRORS IN ALLOWABLE RANGE
        GOTO 700
690     ERRCNT(I)=0
695     CHEK1(I)=0
        CHEK2(I)=0
700     CONTINUE
C
C     *****C170 CONTROLLERS******
C     *****RAMP IS CHECKED EVERY SEC,ALL OTHERS EVERY 2 SECS*****
        K=0
        DO 800 I=1,NRAMPS
        IF(I.GE.3.AND.I.LE.7)GOTO 800 ;CN710 CONTROLLERS
        K=K+1
        LMT=30
        IF(I.EQ.10)GOTO 810 ;C170 CONTROLLER WITH 1 RAMP.CHECK EACH SEC
        IF(NBRAMP.EQ.1)GOTO 800 ;NOT TIME TO CHECK YET
        LMT=15
810     IF(CSTAT(I).EQ.1)GOTO 880 ;ERROR ALREADY NOTED
        IF(CYCOVR(I).EQ.9)GOTO 880 ;CONTROLLER OFFLINE
        CHEK1(I)=ISHFT(RCV170(K),8) ;MOVE LEFT 8 BITS
        CHEK2(I)=XMT170(K).AND.177400K ;RETAIN BITS 8-15
        RCV170(K)=0 ;REINIT RECEIVED BUFFER
        IF(ITEST(CHEK1(I),13))GOTO 850 ;RETURN MESSAGE BIT 5 ON ?
        ERRBS(K)=0
830     IF(CHEK1(I).EQ.CHEK2(I))GOTO 890
        ERRCNT(I)=ERRCNT(I)+1 ;INC ERROR COUNTS
        IF(ERRCNT(I).LT.LMT)GOTO 895
        GOTO 800
850     ERRBS(K)=ERRBS(K)+1 ;INCREMENT ERROR COUNT
        IF(ERRBS(K).GE.LMT)GOTO 880
        CALL ICLR(CHEK1(I),13)
        GOTO 830
880     ERRBS(K)=0
890     ERRCNT(I)=0
895     CHEK1(I)=0
        CHEK2(I)=0
800     CONTINUE
        RETURN
        END
```

```
      OVERLAY ORAMPS
      SUBROUTINE RAMPS
C
C
C FUNCTION : TO COLOUR THE VECTORED RAMPS FOR THE Q DISPLAY AND THE RAMP
C            POLYGONS FOR THE INTERCHANGE DISPLAYS. THE ROUTINE DETERMINES
C            WHICH MAP IS TO BE COLOURED AND BY MEANS OF POINTERS THE
C            THE APPROPRIATE POLYGON IS WRITTEN TO THE DISPLAY.
C      THE COLOUR IS BASED UPON QUEUE CONDITIONS WHICH ARE
C    PASSED BY THE LOGICAL VARIABLES NOTIN INAB,INBL.
C
C
      COMMON/GRAPH/MAPNUM,COLOR
      COMMON/QDOR/NOTIN,INBL,INAB
      COMMON/SCREEN/GCHN,PFILE,RQEN,RNTEMP
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/CREST/M,N
      COMMON/WORDS/NSTA,NCOM,IS,IV,IO,RO,RV,TXT,K,K1
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER M,N
      INTEGER COLOR,IDATA(39),GCHN,PFILE,RQEN,RNTEMP
      LOGICAL NOTIN(10),INAB(10),INBL(10)
C
C
C     CHECK IF DISPLAYING AN INTERCHANGE MAP.IF SO SKIP TO 30
C
      IF(MAPNUM .GT. 1) GOTO 30
C
C      M & N REPRESENT THE NUMBER OF RAMPS  IN QEWDYN.
C     NPOLY IS THE NUMBER OF POLYGONS REQ'D TO DISPLAY A GIVEN RAMP
C
C     M A I N   D I S P L A Y
C
      DO 25 I=1,NRAMPS
      IF(.NOT.NOTIN(I)) WRITE(GCHN,200)
      IF(.NOT.NOTIN(I)) GOTO 35
      IF(INBL(I)) WRITE(GCHN,210)
      IF(INAB(I)) WRITE(GCHN,220)
35    READ(PFILE,18) (IDATA(J),J=1,39)
18    FORMAT(39A2)
37    FORMAT(' ',39A2)
      WRITE(GCHN,37) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+000046K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 35
C
C     RAMPS UNDER A BRIDGE REQUIRE 2 VECTOR COMMANDS
C
      IF(I.EQ. 4 .OR. I.EQ. 9) GOTO 20
      GOTO 25
20    READ(PFILE,18) (IDATA(J),J=1,39)
      WRITE(GCHN,37) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 20
25    CONTINUE
      RETURN
C
C     I N T E R C H A N G E   D I S P L A Y S
C     READ FILE TO DETERMINE ELEMENT NUMBERS FOR THE INTERCHANGE
C     RAMP IN QUESTION
C
C
C     READ FILE TO DETERMINE ELEMENT NUMBERS FOR THE INTERCHANGE RAMP
C
30    READ(PFILE,27)M,N
27    FORMAT(2I2)
      DO 100 I=M,N
      IF(.NOT. NOTIN(I)) WRITE(GCHN,200)   ;FALSE IMPLIES NOT IN QOR
      IF(.NOT. NOTIN(I)) GOTO 55
      IF(INBL(I)) WRITE(GCHN,210)
      IF(INAB(I)) WRITE(GCHN,220)
C
C     INITIALIZE THE NUMBER OF POLYGONS TO DISPLAY A GIVEN RAMP
C
C     RAMPS UNDER BRIDGES REQUIRE 2 POLYGONS IE. CANTHRA SR
C                                                HWY 10 SR
C                                                WINSTON SR
C
55    NPOLY=1
      IF(I.EQ.2.OR.I.EQ.4 .OR.I.EQ.9) NPOLY=2
C
C     READ RAMP POLYGONS FROM FILE
C
      DO 60 IPOLY=1,NPOLY
60    READ(PFILE,18) (IDATA(J),J=1,39)
```

```
      WRITE(ICHAN,37) (IDATA(J),J=1,39)
      KK=ISHFT(IDATA(39),8)+66K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 60
60    CONTINUE
100   CONTINUE
200   FORMAT(" ",":"|COL C2 C0")
210   FORMAT(" ",":"|COL C4 C0")
220   FORMAT(" ",":"|COL C1 C0")
      RETURN
      END
      OVERLAY ONCTIME
      SUBROUTINE RCTIME
      DIMENSION SOCCIM(22),SVOLIM(22)
      DIMENSION RAMTAB(40),CYCTAB(25)
      DIMENSION LOOCC(10)
      COMMON/QO1/QOVR(10),QIND(10)
      COMMON/QO2/QWAIT(10),ULNTH(10)
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/CTRL5/THRCP(4),CTMMOD
      COMMON/WKDAY/HMS(3),MDY(3)
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/D30/DATA30(710)
      COMMON/QOC/QOCC(13)
      COMMON/RUET/NPO,NOO,NEO,NOD
      INTEGER SOCCIM,SVOLIM
      INTEGER RAMTAB,CYCTAB
      INTEGER QOVR,QIND
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER CYCOVR,RAMCYC
      INTEGER THRCP,CTMMOD
      INTEGER HMS,MDY
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER DATA30
      INTEGER INDEX,ID,K
      INTEGER LOCC,DOCC,VOL,CYC1,CYC2,CYC3,CYC
      INTEGER IHRMIN
      INTEGER QWAIT,ULNTH,QOCC
      EQUIVALENCE (SOCCIM(1),DATA30(587)),(SVOLIM(1),DATA30(609))
      EQUIVALENCE (LOOCC(1),DATA30(126))
C
C
      CALL OPEN(5,"RAMTAB",2,IER,2*40)
      CALL READR(5,1,RAMTAB,1,IER)
      CALL CLOSE(5,IER)
      CALL OPEN(5,"CYCTAB",2,IER,2*25)
      CALL READR(5,1,CYCTAB,1,IER)
      CALL CLOSE(5,IER)
C
C
C DETERMINE CONTROL PERIOD
C
      IF(CTMMOD.GE.4)GOTO 50 ;AUTO ADVANCEMENT IN CONTROL PERIOD
      CONTRL=CTMMOD ;MANUAL C.P. FROM OPERATOR
      GOTO 70
50    CONTRL=0
      IHRMIN=HMS(1)*100+HMS(2)
      DO 60 I=1,4
60    IF(IHRMIN.GE.THRCP(I))CONTRL=MOD(I,4)
C
70    CONTINUE
      IF(CONTRL.EQ.0)GOTO 400 ;CP=0,NO NEED TO CALCULATE ANYTHING
C
      DO 100 I=1,NRAMPS
      IF(CONTRL.EQ.3.AND.METER(I).EQ.0)GOTO 100 ;CONTROLLER OFFLINE
      IF(QOVR(I).EQ.1)QIND(I)=0 ;QOVR DISABLED
      IF(QIND(I).EQ.1)GOTO 100 ;QOVR ON
      IF(CYCOVR(I).NE.0)GOTO 100 ;MANUAL OVERRIDE FOR THIS RAMP
      INDEX=4*I-3
      ID=RAMTAB(INDEX) ;LOCAL STATION ID
      LOCC=SOCCIM(ID) ;LOCAL STN 1-MIN OCC
      ID=RAMTAB(INDEX+1) ;DOWNSTR STN ID
      DOCC=SOCCIM(ID) ;DOWNSTR STN 1-MIN OCC
      ID=RAMTAB(INDEX+2) ;UPSTR STN ID
      VOL=SVOLIM(ID) ;UPSTR STN 1-MIN VOL
      ID=RAMTAB(INDEX+3) ;CONTROL SECTION ID
C
C CYCLE LENGTH CALCULATIONS BASED ON MAINLINE CONDITIONS
C LOCAL OCC,DOWNSTREAM OCC,AND UPSTREAM VOL
C
      CYC1=0
      CYC2=0
      CYC3=0
      K=(ID-1)*MAXCYC
      DO 200 J=1,MAXCYC
      IF(LOCC.LT.CYCTAB(J))GOTO 210
      CYC1=J ;CYC LNTH BASED ON LOCAL OCC
210   IF(DOCC.LT.CYCTAB(J+MAXCYC))GOTO 220
      CYC2=J ;CYC LNTH BASED ON DOWNSTR OCC
220   IF(VOL.LT.CYCTAB(K+J))GOTO 200
      CYC3=J ;CYC LNTH BASED ON UPSTREAM VOL
200   CONTINUE
      CYC=MAX0(CYC1,CYC2,CYC3) ;PICK THE LARGEST VALUE
      IF(CONTRL.EQ.1.AND.METER(I).EQ.1.AND.CYC.EQ.0)CYC=1;MIN CYC,MET ON
      IF(CONTRL.EQ.2.AND.CYC.EQ.0)CYC=1;MIN CYC IN CP2
```

```
C
C CONTROL MODES
C
      IF(CYC1.GT.CYC2.AND.CYC1.GT.CYC3)MODCYC(I)=1 ;LOCAL OCC CONTROL
      IF(CYC2.GT.CYC1.AND.CYC2.GT.CYC3)MODCYC(I)=2 ;DOWNSTR OCC
      IF(CYC3.GT.CYC2.AND.CYC3.GT.CYC1)MODCYC(I)=3 ;UPSTR VOL
      IF(CYC1.EQ.CYC2.AND.CYC1.GT.CYC3)MODCYC(I)=6 ;LOCAL AND DOWNS OCC
      IF(CYC2.EQ.CYC3.AND.CYC2.GT.CYC1)MODCYC(I)=7 ;DOCC AND UPVOL
      IF(CYC3.EQ.CYC1.AND.CYC3.GT.CYC2)MODCYC(I)=8 ;LOCC AND UPVOL
      IF(CYC1.EQ.CYC2.AND.CYC1.EQ.CYC3)MODCYC(I)=9 ;LOCC.DOCC.UPVOL
      RAMCYC(I)=CYC  ;CYC LNTH OUTPUT
      METER(I)=1 ;METERING ON FOR POS CYC LNTH
      IF(CYC.EQ.0)METER(I)=0 ;NO METERING
100   CONTINUE
      GO TO 471
460   DO 470 I=1,NRAMPS
      IF([illegible](I).NE.0)GOTO 470 ;THIS RAMP UNDER MANUAL OVERRIDE
      RAMCYC(I)=0 ;P GREEN
      METER(I)=0 ;NO METERING
470   CONTINUE
471   DO 900 I=1,NVD
      IF(DOCC(I).LT.0.AND.LDOCC(I).GE.0)DOCC(I)=0
      DOCC(I)=(6*DOCC(I)+4*LDOCC(I))/10
900   IF(LDOCC(I).LT.0)DOCC(I)=-1
C
      DO 700 I=1,NRAMPS
      QLNTH(I)=0
700   QWAIT(I)=0 ;INIT QLNTH & WAIT TIME TO 0
      RETURN
      END
      OVERLAY URDISMAP
      SUBROUTINE RDISMAP
C
C FUNCTION : TO DISPLAY NON DESCRIPTIVE GRAPHICS FOR INTERCHANGE DISPLAYS
C             -CHECKS FOR MISCELANEOUS POLYGONS & DISPLAYS IN GREEN
C             -DETERMINES THE NUMBER OF DETECTOR STATIONS AND DISPLAYS
C             -DISPLAYS RAMP METERING ARROWS
C             -LEGEND
C             -TEXTUAL DATA FOR CLARITY
C             -NORTHPOINT
C
C
      COMMON/SCREEN/GCHN,PFILE,RGEN,RNTEMP
      COMMON/OCCHN/OCC,RMETER
      COMMON/CREST/M,N
      COMMON/WORDS/NSTA,NCON,IS,IV,ID,RO,RV,TXT,K,K1
C
      INTEGER NSTA,NCON,IS,IV,ID,RO,RV,TXT(6),K,K1
      INTEGER IDATA(39)
      INTEGER GCHN,PFILE,RGEN,RNTEMP
C
      LOGICAL RMETER(10),OCC
C
C     CHECK FOR MISCELANEOUS POLYGONS AND DISPLAY IN GREEN
C
      READ(PFILE,100) MISPOLY
      IF(MISPOLY .GT. 0) WRITE(GCHN,250)
      IF(MISPOLY .GT. 0) GOTO 125
      GOTO 130
125   DO 200 J=1,MISPOLY
127   READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+40K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 127
200   CONTINUE
250   FORMAT(" ","ICOL C2 C0")
C
C     READ TOTAL NUMBER OF DETECTORS FOR THIS DISPLAY
C     DISPLAY DETECTOR STATIONS FOR INTERCHANGES WHITE ON BLACK
C
130   READ(PFILE,110) NDP
100   FORMAT(I1)
110   FORMAT(I2)
      WRITE(GCHN,120)
120   FORMAT(" ","ICOL C0 C7")
      DO 140 I=1,NDP
      READ(PFILE,10) (IDATA(J),J=1,39)
10    FORMAT(39A2)
      WRITE(GCHN,35) (IDATA(J),J=1,39)
35    FORMAT(" ",39A2)
140   CONTINUE
C
C     DISPLAY RAMP METERING ARROW (TRUE=RED   FALSE=GREEN)
C
      DO 150 I=M,N
      IF(RMETER(I)) WRITE(GCHN,300)
      IF(.NOT. RMETER(I)) WRITE(GCHN,310);      FALSE IMPLIES NOT METERED
      READ(PFILE,10) (IDATA(J),J=1,39)
      WRITE(GCHN,35) (IDATA(J),J=1,39)
150   CONTINUE
C
C     DISPLAY ANY ADDITIONAL DATA
C
5     READ(PFILE,10,END=99) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      GOTO 5
310   FORMAT(" ","ICOL C2 C7")
300   FORMAT(" ","ICOL C1 C7")
C
99    RETURN
      END
      OVERLAY OREGEN
      SUBROUTINE REGEN
C
C  FUNCTION: THIS ROUTINE READS THE GENDYN DATA FILE AND PASSES THE
C            DATA BACK TO THE MAIN GRAPHIC PROGRAM.
C
C  CALLED BY:TASK GMAP
C
      COMMON/SCREEN/GCHN,PFILE,RGEN,NHTEMP
      COMMON/NREC/KL300,NNREC,KINC,KSM,NALREC,KDISP
      COMMON/GENDATA/GEN
C
      INTEGER GCHN,PFILE,RGEN,NHTEMP
      INTEGER KL300,NNREC,KINC,KSM,NALREC,KDISP
      INTEGER GEN(512),SREC
C
      CALL OPEN(RGEN,"GENDYN",1,IER,128)
      SREC=KDISP-8
      CALL READR(RGEN,SREC,GEN(1),8,IER)
      CALL CLOSE(RGEN,IER)
C
      RETURN
      END
      SUBROUTINE SLCHEK(VOL,OCC,AEL,NVE,IJ)
C
      COMMON/LIMITS/MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      COMMON/STTS/LATST,ERMAND
      COMMON/RANGE/UPRNGE(113,2),DNRNGE(67,3)
C
      INTEGER MAXOCC,MAXSPD,IMIN,MAXVEH,STDLN,S1,S2
      INTEGER LATST,ERMAND
      INTEGER UPRNGE,DNRNGE
      INTEGER VOL,OCC,AEL,NVE,IJ,IROW
C
C
      IROW=1
      IF(VOL.GT.MAXVEH.OR.VOL.LT.0)GOTO 120 ;VOL OUT OF RANGE
      IF(OCC.LT.0)GOTO 130 ;OCC OUT OF RANGE
      IF(OCC.GT.MAXOCC)OCC=MAXOCC
      IF(VOL.EQ.0.AND.OCC.EQ.0)NVE=1
      AEL=AEL+OCC ;ACC EL TIME
      IF(VOL.EQ.0)GO TO 999

      IF(AEL.GE.(VOL*IMIN))GOTO 100
      IROW=2
120   UPRNGE(IJ,IROW)=UPRNGE(IJ,IROW)+1 ;INC OUT-OF-RANGE COUNTS
130   ERMAND=ERMAND+1 ;INC ERROR COUNTS
      VOL=0
      OCC=0
100   AEL=0
C
999   RETURN
      END
      OVERLAY OSPD1
      SUBROUTINE SPD1
C
C
C FUNCTION : DETERMINE WHAT COLOUR TO PAINT THE LANES BASED UPON
C         C1=RED  C2=GREEN  C3=BLUE  C4=YELLOW  C5=ORANGE
C            THE 1 MINUTE MAINLANE DATA, ACCORDING TO SPEED.
C
C
      COMMON/QEMDATA/QEM
      COMMON/SCREEN/GCHN,PFILE,NQEM,RWTEMP
      COMMON/OPTA/ISOL,ISCL,ISPL
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1,IDATA(39)
      INTEGER QEM(512),GCHN,PFILE,NQEM,RWTEMP
C
C     TEST SPEED ON PASSING LANE
C
      IF(QEM(ISPL) .GT. 100) WRITE(GCHN,300)
      IF(QEM(ISPL) .GE. 80 .AND. QEM(ISPL) .LE. 100) WRITE(GCHN,200)
      IF(QEM(ISPL) .GE. 60 .AND. QEM(ISPL) .LT. 80) WRITE(GCHN,400)
      IF(QEM(ISPL) .GE. 40 .AND. QEM(ISPL) .LT. 60) WRITE(GCHN,500)
      IF(QEM(ISPL) .LT. 40) WRITE(GCHN,100)
      IF(QEM(ISPL) .LT. 0) WRITE(GCHN,600)
C
C     PAINT PASSING LANE
C
900   READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 900
C
C     TEST SPEED ON CENTRE LANE
C
      IF(QEM(ISCL) .GT. 100) WRITE(GCHN,300)
      IF(QEM(ISCL) .GE. 80 .AND. QEM(ISCL) .LE. 100) WRITE(GCHN,200)
      IF(QEM(ISCL) .GE. 60 .AND.QEM(ISCL) .LT. 80) WRITE(GCHN,400)
      IF(QEM(ISCL) .GE. 40 .AND. QEM(ISCL) .LT. 60) WRITE(GCHN,500)
      IF(QEM(ISCL) .LT. 40) WRITE(GCHN,100)
      IF(QEM(ISCL) .LT. 0) WRITE(GCHN,600)
C
C       PAINT CENTRE LANE
C
1000  READ(PFILE,10) (IDATA(I),I=1,39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      KK=ISHFT(IDATA(39),8)+46K
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 1000
      RETURN
10    FORMAT(39A2)
35    FORMAT(" ",39A2)
100   FORMAT(" ","|COL C1 C0")
200   FORMAT(" ","|COL C2 C0")
400   FORMAT(" ","|COL C4 C0")
500   FORMAT(" ","|COL C5 C0")
300   FORMAT(" ","|COL C3 C0")
600   FORMAT(" ","|COL C7 C0")
      END
      OVERLAY USPD2
      SUBROUTINE SPD2
C
C
C FUNCTION 1 TO PAINT DRIVING LANES AND EXTRA LANE ON WINSTON CHURCHILL
C           BASED UPON SPEED.
C
C
      COMMON/GRAPH/MAPNUM,COLOR
      COMMON/QEWDATA/QEW
      COMMON/SCREEN/GCHN,PFILE,RQEW,RWTEMP
      COMMON/OPTA/ISDL,ISCL,ISPL
      COMMON/EXTRA/WINSTON
      COMMON/WORDS/NSTA,NCON,IS,IV,IO,RO,RV,TXT,K,K1
      LOGICAL WINSTON
      INTEGER NSTA,NCON,IS,IV,IO,RO,RV,TXT(6),K,K1
          INTEGER QEW(512),GCHN,PFILE,RQFW,RWTEMP,COLOR,IDATA(39)
C
C     TEST SPEED ON DRIVING LANE
C
      IF(QEW(ISDL) .GT. 100) WRITE(GCHN,300)
      IF(QEW(ISDL) .GE. 80 .AND. QEW(ISDL) .LE. 100) WRITE(GCHN,200)
      IF(QEW(ISDL) .GE. 60 .AND. QEW(ISDL) .LT. 80) WRITE(GCHN,400)
      IF(QEW(ISDL) .GE. 40 .AND. QEW(ISDL) .LT. 60) WRITE(GCHN,500)
      IF(QEW(ISDL) .LT. 40) WRITE(GCHN,100)
      IF(QEW(ISDL) .LT. 0) WRITE(GCHN,800)
C
C       PAINT DRIVING LANE
C
900   READ(PFILE:10) (IDATA(I):I=1:39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
      [illegible]
      IF(KK .EQ. K .OR. IDATA(37) .EQ. K1) GOTO 900
      IF(MAPNUM .NE. 3) RETURN
      IF(WINSTON) RETURN
C
C     TEST SPEED ON WINSTON EXTRA LANE
C
      ISEL=612
      IF(QEW(ISEL) .GT. 100) WRITE(GCHN,300)
      IF(QEW(ISEL) .GE. 80 .AND. QEW(ISEL) .LE. 100) WRITE(GCHN,200)
      IF(QEW(ISEL) .GE. 60 .AND. QEW(ISEL) .LT. 80) WRITE(GCHN,400)
      IF(QEW(ISEL) .GE. 40 .AND.QEW(ISEL) .LT. 60)WRITE(GCHN,500)
      IF(QEW(ISEL) .LT. 40) WRITE(GCHN,100)
      IF(QEW(ISEL) .LT. 0) WRITE(GCHN,800)
C
C       PAINT EXTRA LANE
C
      READ(PFILE:10) (IDATA(I):I=1:39)
      WRITE(GCHN,35) (IDATA(I),I=1,39)
C
C     SET WINSTON=.TRUE. AFTER PAINTING EXTRA LANE
C
      WINSTON=.TRUE.
      RETURN
15    FORMAT(39A2)
35    FORMAT(' ',39A2)
100   FORMAT(' ','[COL C1 C0')
200   FORMAT(' ','[COL C2 C0')
400   FORMAT(' ','[COL C4 C0')
500   FORMAT(' ','[COL C5 C0')
300   FORMAT(' ','[COL C3 C0')
800   FORMAT(' ','[COL C7 C0')
      END
```

```
C     OVERLAY OT12A
      SUBROUTINE T12A (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,FLAG)
C
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/CTRL3/THRCP(4),CTRMOD
C
      INTEGER ICODE(NVAR),INUM(ICOUNT),ICHAR(ICHCNT)
      INTEGER CYCOVR,RAMCYC,THRCP,CTRMOD,FLAG,CONTRL
C
C
C
      FLAG=1
      IF(ICHAR(2).NE.'  ')GOTO 1
      IF(ICHAR(1).EQ.'C ')GOTO 100
      IF(ICHAR(1).EQ.'P ')GOTO 200
      IF(ICHAR(1).EQ.'T ')GOTO 300
      IF(ICHAR(1).EQ.'R ')GOTO 400
1     TYPE "COMMAND NOT AUTHORIZED!"
      GOTO 1000
2     TYPE "DATA ERROR."
      GOTO 1000
C     COMMAND : C
100   IF(NVAR.EQ.1)GOTO 130
      IF(ICHCNT.NE.4)GOTO 1
      IF(ICOUNT.LT.2)GOTO 1
      IF(ICHAR(3).NE.'M ')GOTO 1
      IF(ICODE(NVAR-1).NE.2)GOTO 1
      IF(INUM(ICOUNT).GT.20)GOTO 2
      J=ICOUNT-1
      DO 110 I=1,J
      IF(INUM(I).LT.1)GOTO 2
      IF(INUM(I).GT.10)GOTO 2
110   CONTINUE
      DO 120 I=1,J
120   CYCOVR(INUM(I))=INUM(ICOUNT)
      GOTO 1000
130   WRITE(10,135)CYCOVR
135   FORMAT(' ',10I4)
      GOTO 1000
C     COMMAND : P
200   IF(NVAR.EQ.1)GOTO 210
      IF(NVAR.NE.2)GOTO 1
      IF(ICOUNT.NE.1)GOTO 1
      IF(INUM(1).GT.4)GOTO 2
      CTRMOD=INUM(1)
      GOTO 1000
210   IF(CTRMOD.EQ.0)GOTO 220
      WRITE(10,215)CTRMOD
215   FORMAT(' ','PRESENT CP=',I1,', MANUAL.')
      GOTO 1000
220   WRITE(10,225)CONTRL
225   FORMAT(' ','PRESENT CP=',I1,',AUTO ADVANCEMENT.')
      GOTO 1000
C     COMMAND : T
300   IF(NVAR.EQ.1)GOTO 330
      IF(NVAR.NE.3)GOTO 1
      IF(ICOUNT.NE.4)GOTO 1
      DO 310 I=1,4
      IF(INUM(I).GE.2400)GOTO 2
      IF(MOD(INUM(I),100).GT.59)GOTO 2
      IF(INUM(I-1).GT.INUM(I))GOTO 2
310   CONTINUE
      DO 320 I=1,4
320   THRCP(I)=INUM(I)
      GOTO 1000
330   WRITE(10,340)THRCP
340   FORMAT(/1X,'CP START TIME: ',4(I4,2X))
      GOTO 1000
C     COMMAND : R
400   FLAG=2
C
1000  RETURN
      END
```

```
C
      OVERLAY OT128
      SUBROUTINE T128 (INUM,ICOUNT,ICHCNT)
C
      INTEGER INUM(0:ICOUNT),ICHCNT
      INTEGER DUMMY(50)
C
C
C
      IF(ICHCNT.NE.2)GOTO 1
      IF(ICOUNT.LT.1)GOTO 1
      IF(INUM(1).EQ.0)GOTO 2
      IF(INUM(1).GT.3)GOTO 2
      ISUB=INUM(1)
      GOTO(100,200,300),ISUB
100   CALL OPEN(6,"RANTAB",2,IER,80)
      CALL READR(6,1,DUMMY,1,IER)
      IF(ICOUNT.NE.1)GOTO 110
      WRITE(10,105)(DUMMY(I),I=1,40)
105   FORMAT(/1X,"RANTAB:",10(/1X,4I6))
      GOTO 1000
110   MAXVAR=7 ; MAXIMUM # OF VARIABLES
      MAXNUM=10 ; NUMBER OF ROWS
      ICOL=4 ; NUMBER OF COLUMNS
      GOTO 400
200   CALL OPEN(6,"CYCTAB",2,IER,50)
      CALL READR(6,1,DUMMY,1,IER)
      IF(ICOUNT.NE.1)GOTO 210
      WRITE(10,205)(DUMMY(I),I=1,25)
205   FORMAT(/1X,"CYCTAB:",5(/1X,5I6))
      GOTO 1000
210   MAXVAR=8
      MAXNUM=5
      ICOL=5
      GOTO 400
300   CALL OPEN(6,"CYCLES",2,IER,100)
      CALL READR(6,1,DUMMY,1,IER)
      IF(ICOUNT.NE.1)GOTO 310
      WRITE(10,305)(DUMMY(I),I=1,50)
305   FORMAT(/1X,"CYCLES:",10(/1X,5I6))
      GOTO 1000
310   MAXVAR=8
      MAXNUM=10
      ICOL=5
400   IF(NVAR.NE.MAXVAR)GOTO 1
      IF(INUM(2).GT.MAXNUM)GOTO 2
      IF(INUM(2).EQ.0)GOTO 2
      J=INUM(2)*ICOL-ICOL
      DO 410 I=1,ICOL
410   DUMMY(I+J)=INUM(I+2)
      CALL WRITR(6,1,DUMMY,1,IER)
      GOTO 1000
1     TYPE "COMMAND NOT AUTHORIZED!"
      GOTO 1000
2     TYPE "DATA ERROR."
      GOTO 1000
1000  CALL CLOSE(6,IER)
C
      RETURN
      END
```

```
        OVERLAY OT13
        SUBROUTINE T13 (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
C
        COMMON/QU1/QOVR(10),QIND(10)
        COMMON/QU2/QWAIT(10),QLNTH(10)
        COMMON/QU3/KQCNT(10),KSTEP(10),KTLMT(10),KTHR
        COMMON/QU4/QOCCLB(10),QOCCUB(10)
        COMMON/QU5/QOCC(13)
        COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
C
        INTEGER ICODE(NVAR),INUM(0:ICOUNT),ICHAR(ICHCNT)
        INTEGER OUTBUF(74),QOVR,QIND,QWAIT,QLNTH,KQCNT,KSTEP,KTLMT,KTHR
        INTEGER QOCCLB,QOCCUB,QOCC,CONTRL,MAXCYC,METER,MODCYC
C
C
C
        IF(ICHAR(2).NE.'  ')GOTO 1
        IF(ICHAR(1).EQ.'D ')GOTO 100
        IF(ICHAR(1).EQ.'N ')GOTO 100
        IF(ICHAR(1).EQ.'U ')GOTO 300
        IF(ICHAR(1).EQ.'L ')GOTO 300
        IF(ICHAR(1).EQ.'T ')GOTO 500
        IF(ICHAR(1).EQ.'P ')GOTO 600
1       TYPE "COMMAND NOT AUTHORIZED!"
        GOTO 1000
2       TYPE "DATA ERROR."
        GOTO 1000
C       COMMAND : D AND N
100     IF(NVAR.NE.2)GOTO 1
        IF(ICODE(2).NE.1)GOTO 1
        IF(INUM(1).LT.1)GOTO 2
        IF(INUM(1).GT.10)GOTO 2
        QOVR(INUM(1))=(.NOT.ICHAR(1).AND.1000K)/1000K  ; 1 OR 0
        GOTO 1000
C       COMMAND : U AND L
300     IF(NVAR.NE.11)GOTO 1
        IF(ICOUNT.NE.10)GOTO 1
        DO 320 I=1,10
        IF(ICHAR(1).EQ.'L ')GOTO 312
        QOCCUB(I)=INUM(I)
        GOTO 320
312     QOCCLB(I)=INUM(I)
320     CONTINUE
        GOTO 1000
C       COMMAND : T
500     IF(NVAR.NE.13)GOTO 1
        IF(ICHCNT.NE.4)GOTO 1
        IF(ICODE(1).NE.1)GOTO 1
        IF(ICHAR(3).NE.'S ')GOTO 1
        IF(ICHAR(4).NE.'  ')GOTO 1
        DO 510 I=4,11
        IF(ICODE(I).NE.1)GOTO 1
510     CONTINUE
        DO 520 I=1,10
520     KSTEP(I)=INUM(I+1)
        KTHR=INUM(11)*2
        GOTO 1000
C       COMMAND : P
600     IF(NVAR.NE.1)GOTO 1
        DO 610 I=1,10
        OUTBUF(I)=MODCYC(I)
        OUTBUF(I+10)=QWAIT(I)
        OUTBUF(I+20)=QOVR(I)
        OUTBUF(I+30)=QOCCLB(I)
        OUTBUF(I+40)=QOCCUB(I)
        OUTBUF(I+50)=KSTEP(I)
610     OUTBUF(I+60)=QOCC(I)
        DO 620 I=11,13
620     OUTBUF(I+60)=QOCC(I)
        OUTBUF(74)=(KTHR+1)/2 ; ROUND OFF
        WRITE(10,625)(I,I=1,9),OUTBUF
625     FORMAT(/1X,'QAMPS',//4X,9(2X,'NO.',I1),2X,'NO.10',//1X,'MODCYC ',
     +  10I6,//1X,'QWAIT  ',10I6,//1X,'QOVR   ',10I6,//1X,
     +  'QOCCLB ',10I6,//1X,'QOCCUB ',10I6,//1X,'KSTEP  ',
     +  10I6,//1X,'QOCC',4X,13I5,//1X,'KTHR(MIN.)=',I6)
C
1000    RETURN
        END
```

```
C
        OVERLAY UT14A
        SUBROUTINE T14A (ICODE,ICHAR,NVAR,ICHCNT,FLAG)
C
        COMMON/SGN1/XMTBUF(45),RCVBUF(45),SGNADR(2)
        COMMON/SGN2/NWORDS,BUFSIZ,NMSGTX,NMSGRX,NTLSGN(2)
C
        INTEGER XMTBUF,RCVBUF,SGNADR,ICODE(NVAR),ICHAR(ICHCNT)
        INTEGER MSGBUF(45),BUFSIZ,FLAG,PARBUF(128)
C
C
C
        FLAG=1
        IF(ICHAR(2).NE.'  ')GO TO 1
        IF(ICHAR(1).EQ.'A ')GO TO 100
        IF(ICHAR(1).EQ.'D ')GO TO 5
        IF(ICHAR(1).EQ.'C ')GO TO 5
        IF(ICHAR(1).EQ.'B ')GO TO 5
        IF(ICHAR(1).EQ.'P ')GO TO 5
        IF(ICHAR(1).NE.'S ')GO TO 1
        FLAG=3
        GO TO 1000
C
5       FLAG=2
        GO TO 1000
C
1       TYPE "COMMAND NOT AUTHORIZED!"
        GO TO 1000
C
100     IF(NVAR.NE.1)GO TO 1
C
C       'A' COMMAND
C
        CALL OPEN(6,"MSGFILE",1,IER,90)
        CALL READR(6,1,MSGBUF(1),1,IER)
        NMAX=MSGBUF(1)   ;POSITION OF LAST RECORD
        NEXT=MSGBUF(2)   ;POSITION OF 1ST EMPTY RECORD
        N1=NEXT
110     IF(N1.GT.NMAX)GO TO 120
        CALL READR(6,N1,MSGBUF(1),1,IER)
        IF(MSGBUF(1).EQ.-1)GO TO 130  ;-1 SIGNIFIES EMPTY RECORD
        N1=N1+1
        GO TO 110
120     NMAX=NMAX+1
130     NEXT=N1+1
        CALL CLOSE(6,IER)
C
140     WRITE(10,143)
143     FORMAT(' ',' ROW1,CR;ROW2,CR' / ' ',' 1234567890123456789012')
        READ(11,145)(MSGBUF(I),I=3,13)
145     FORMAT(11A2)
        READ(11,145)(MSGBUF(I),I=15,25)
        WRITE(10,148)
148     FORMAT(' ','OK?  TYPE Y OR N >', Z)
        READ(11,149)IN
149     FORMAT(A1)
        IF(IN.EQ.'N ')GO TO 140
C
        MSGBUF(1)=000044K        ;NUL (000K) / ADDRESS (044K)
        MSGBUF(2)=000407K        ;SOH (001K) / BEL (007K)
        MSGBUF(14)=006412K       ;CR  (015K) / LF (012K)
        MSGBUF(26)=001400K       ;ETX (003K) / NUL (000K)
C
C       WRITE (10, 222) (MSGBUF(I), I=1,45)
C222    FORMAT (' ',' MSGBUF IS' /
C     *         (1X, 5(O16, 'K') ) )
C
        CALL OPEN(6,"PARTAB",1,IER,256)
        CALL READR(6,1,PARBUF(1),1,IER)
        CALL CLOSE(6,IER)
C
C       SET EVEN PARITY FOR MESSAGE HEADER AND TEXT
C
        DO 160 I=2,NWORDS        ;NWORDS=26
        IBYTE=MSGBUF(I)
        ILFT=ISHFT(IBYTE,-8)
        ILFT=PARBUF(ILFT+1)
        IRIT=IBYTE.AND.000377K
        IRIT=PARBUF(IRIT+1)
        MSGBUF(I)=ISHFT(ILFT,8) .OR. IRIT
160     CONTINUE
C
        DO 170 I=27,45
170     MSGBUF(I)=000000K        ;NUL (000K) / NUL (000K)
C
```

```
C       WRITE (10, 222) (MSGBUF(I), I=1,45)
        CALL OPEN(6,"MSGFILE",2,IER,90)
        CALL WRITR(6,N1,MSGBUF(1),1,IER)
        N1=N1-2
        WRITE(10,175)N1,(MSGBUF(I),I=3,13),(MSGBUF(I),I=15,25)
175     FORMAT(' ',"MESSAGE #",I2,':', 2(/12,11A2))
        CALL READR(6,1,MSGBUF(1),1,IER)
        MSGBUF(1)=NMAX
        MSGBUF(2)=NEXT
        CALL WRITR(6,1,MSGBUF(1),1,IER)
        CALL CLOSE(6,IER)
C
1000    RETURN
        END
```

```
C
      OVERLAY UT148
      SUBROUTINE T148 (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
C
C
      COMMON/SGN1/XMTBUF(45),RCVBUF(45),SGNADR(2)
      COMMON/SGN2/NWORDS,BUFSIZ,NMSGTX,NMSGRX,MLTSGN(2)
C
      INTEGER ICODE(NVAR),INUM(ICOUNT),ICHAR(ICHCNT)
      INTEGER XMTBUF,RCVBUF,SGNADR
      INTEGER NWORDS,BUFSIZ,NMSGTX,NMSGRX,MLTSGN
      INTEGER MSGBUF(45)
C
C
C
      IC=ICHAR(1)
      IF(IC.NE.'B ' .AND. IC.NE.'L ')  GO TO 300
C
C     'B' AND 'L' COMMANDS
C
      IF(NVAR.NE.1)GO TO 950
      IF(NMSGTX.GT.0)GO TO 100
      TYPE "NO MESSAGES TRANSMITTED"
      GO TO 1000
C
100   IF (IC.NE.'B ')  GO TO 200
C
C     'B' COMMAND
C
      TYPE "TRANSMIT MESSAGE:"
      IF(XMTBUF(2).NE.1280)GO TO 110
      TYPE " 'END'"
      GO TO 120
110   WRITE BINARY (10) (XMTBUF(I),I=3,25), XMTBUF(14)
120   TYPE " "
      TYPE "RECEIVE MESSAGE:"
      IF(NMSGRX.NE.0)GO TO 130
      TYPE "NO MESSAGES RECEIVED"
      GO TO 1000
130   IF(RCVBUF(2).NE.1280)GO TO 140
      TYPE " 'END'"
      GO TO 1000
140   WRITE BINARY (10) (RCVBUF(I),I=3,25), RCVBUF(14)
      GO TO 1000
C
C     'L' COMMAND
C
200   CALL OPEN(6,"XMTTAB",1,IER,90)
      DO 240 I=1,6
      K=0
      WRITE(10,215)I
215   FORMAT(' ',"CMS ",I1)
      DO 236 N=1,NMSGTX
      CALL READR(6,N,MSGBUF(1),1,IER)
      CALL ICLR(MSGBUF(7),7)
      IF(MSGBUF(7).NE.SGNADR(I))GO TO 236
      K=K+1
      IF(MSGBUF(8).EQ.1280)GO TO 230
      WRITE(10,225)(MSGBUF(L),L=4,6),(MSGBUF(L),L=9,19),(MSGBUF(L),L=21,31)
225   FORMAT(/1X,I2,2(":",I2),"-", 2(/1X,11A2))
      GO TO 232
230   WRITE(10,231)(MSGBUF(L),L=4,6)
231   FORMAT(/1X,I2,2(":",I2),//1X,"'END'")
232   IF(K.EQ.NMSGTX)GO TO 900
236   CONTINUE
      IF(K.EQ.0)TYPE "NO MESSAGES TO CMS"
240   CONTINUE
      GO TO 900
C
300   IF (IC.NE.'D ' .AND. IC.NE.'P ')  GO TO 950
C
C     'D' AND 'P' COMMANDS
C
      CALL OPEN(6,"MSGFILE",2,IER,90)
      CALL READR(6,1,MSGBUF(1),1,IER)
      NMAX=MSGBUF(1)
      NEXT=MSGBUF(2)
      N=INUM(1)+2
      IF(N.LE.NMAX)GO TO 320
310   CONTINUE
      N=N-2
      TYPE "MESSAGE #", N, " DOES NOT EXIST"
      GO TO 900
320   CALL READR(6,N,MSGBUF(1),1,IER)
      IF(MSGBUF(1).EQ.-1)GO TO 310
      IF(IC.NE.'D ')  GO TO 400
C
C     'D' COMMAND
C
      IF(INUM(1).NE.0)GO TO 322
      TYPE "YOU CAN'T DELETE 'END'."
      GO TO 900
322   MSGBUF(1)=-1
      CALL WRITW(6,N,MSGBUF(1),1,IER)
      IF(N.NE.NMAX)GO TO 340
330   NMAX=NMAX-1
      CALL READW(6,NMAX,MSGBUF(1),1,IER)
      IF(MSGBUF(1).EQ.-1)GO TO 330
340   MSGBUF(1)=NMAX
      IF(N.LT.NEXT)NEXT=N
      MSGBUF(2)=NEXT
      CALL WRITW(6,1,MSGBUF(1),1,IER)
      GO TO 900
C
C     'P' COMMAND
C
400   IF(INUM(1).NE.0)GO TO 402
      TYPE "MESSAGE #0:"
      TYPE "'END'"
      GO TO 900
402   WRITE(10,405)INUM(1),(MSGBUF(J),J=3,13),(MSGBUF(J),J=15,25)
405   FORMAT(' ','MESSAGE #',I2,': ', 2(/1X,11A2))
C
900   CONTINUE
      CALL CLOSE (6, IER)
      GO TO 1000
C
930   CONTINUE
      TYPE "COMMAND NOT AUTHORIZED!"
C
1000  CONTINUE
      RETURN
      END
```

```
C
      OVERLAY OT14C
      SUBROUTINE T14C (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
C
C
      COMMON/MD/MODENT(299),MODEM(140),MODEM1(73),MUMMDM
      COMMON/SGN1/INTBUF(45),RCVBUF(45),SGNADR(2)
      COMMON/SGN2/NWORDS,BUFSIZ,NMSGTX,NMSGRX,MLTSGN(2)
      COMMON/SGN3/SGNSTAT(2),ALMFLG,KCNTO,P8FLG,AMODE
      COMMON/BLOK1/FT140
C
      INTEGER ICODE(NVAR),INUM(0:ICOUNT),ICHAR(ICHCNT)
      INTEGER SGNSTAT,ALMFLG,KCNTO,P8FLG,AMODE
      INTEGER INTBUF,RCVBUF,SGNADR,BUFSIZ,MLTSGN,MSGBUF(45),MNS,MDY
      INTEGER FT140
C
C
C
C
      IF(NVAR.EQ.2)GO TO 200
C
C     'S' COMMAND
C
      IF(ICHCNT.NE.4)GO TO 1
      IF(ICODE(NVAR-1).NE.2)GO TO 1
      IF(ICHAR(3).NE.'N ')GO TO 1
      IF(ICOUNT.LT.2 .OR. ICOUNT.GT.3)GO TO 1
C
      IEND=ICOUNT-1
      IND=0
      DO 100 I=1,IEND
      INI=INUM(I)
      IF (INI.LT.1 .OR. INI.GT.2) GO TO 2
      IF(SGNSTAT(INI).LT.1)GO TO 100
      TYPE "MALFUNCTION ON SIGN #", INI, "; MESSAGE NOT SENT"
      IND=IND+1
100   CONTINUE
      IF(IND.EQ.IEND)GO TO 1000
C
      IF(MODENT(40).GE.0)GO TO 102
      TYPE "MODENT(40) HARDWARE ERROR."
      GO TO 1000
C
102   CALL OPEN(6,"MSGFILE",1,IER,98)
      CALL READR(6,1,MSGBUF(1),1,IER)
      N=INUM(ICOUNT)+2
      IF(N.LE.MSGBUF(1))GO TO 110
      CALL CLOSE(6,IER)
2     TYPE "DATA ERROR."
      GO TO 1000
1     TYPE "COMMAND NOT AUTHORIZED!"
      GO TO 1000
C
110   CALL READR(6,N,MSGBUF(1),1,IER)
      CALL CLOSE(6,IER)
      IF(MSGBUF(1).NE.-1)GO TO 120
      TYPE "MESSAGE #", INUM(ICOUNT), " DOES NOT EXIST"
      GO TO 1000
C
120   CONTINUE
      DO 125 I=1,45
125   INTBUF(I)=MSGBUF(I)
C
      FT140=1
      GO TO 1000
C
C
200   CONTINUE
      IF(ICOUNT.NE.1)GO TO 1
      IF(INUM(1).LT.1 .OR. INUM(1).GT.2)GO TO 2
      CALL OPEN(6,"AMTTAB",1,IER,98)
      N=NMSGTX
260   CONTINUE
      IF(N.LT.1) GO TO 212
      CALL READR(6,N,MSGBUF(1),1,IER)
      IF(MSGBUF(7).EQ.SGNADR(INUM(1))) GO TO 220
      N=N-1
      GO TU 260
C
212   TYPE "NO MESSAGES TO SIGN"
      GO TO 1000
C
220   CONTINUE
      IF (MSGBUF(6).EQ.1280)
     *   TYPE "'END'"
```

```
      IF (MSGBUF(8).NE.1000)
     *   WRITE BINARY (IO) (MSGBUF(I),I=9,31),MSGBUF(20)
C
900   CONTINUE
      CALL CLOSE(6,IER)
      GO TO 1000
C
1000  CONTINUE
      RETURN
      END
```

```
C
      OVERLAY UT14D
      SUBROUTINE T14D (INUM, ICOUNT)
C
C
      COMMON/TSKSYN/TSK0,TSK1,TSK2,TSK3
      COMMON/SGN1/XMTBUF(45),RCVBUF(45),SGNADR(2)
      COMMON/SGN2/NWORDS,BUFSIZ,NMSGTX,NMSGRX,MTLSGN(2)
      COMMON/SGN3/SGNSTAT(2),ALMFLG,KCNTO,P4FLG,AMODE
      COMMON/BLOK1/FT14D
      COMMON/NRDAT/HMS(3), MDY(3)
C
      INTEGER TSK0,TSK1,TSK2,TSK3
      INTEGER XMTBUF,RCVBUF,SGNADR
      INTEGER NWORDS,BUFSIZ,NMSGTX,NMSGRX,MTLSGN
      INTEGER SGNSTAT,ALMFLG,KCNTO,P4FLG,AMODE
      INTEGER FT14D
      INTEGER HMS ,MDY,INUM(0:ICOUNT)
      INTEGER MSGBUF(45)
C
C
C
      IF (FT14D .NE. 1)  GO TO 900
      FT14D=0
C
      IEND = ICOUNT - 1
      DO 425 I=1,45
425   MSGBUF(I) = XMTBUF(I)
      DO 431 I=2,39
431   MSGBUF(I+6)=XMTBUF(I)
      MSGBUF(1)=MDY(3)
      MSGBUF(2)=MDY(1)
      MSGBUF(3)=MDY(2)
C
      CALL TIME(HMS(1),IER)
      DO 432 I=1,3
432   MSGBUF(I+3)=HMS(I)
C
      CALL OPEN(6,"XMTTAB",3,IER,90)
      DO 436 I=1,IEND
      IF(SGNSTAT(INUM(I)).GT.0)GO TO 436
      MSGBUF(7)=SGNADR(INUM(I))
      NMSGTX=NMSGTX+1
      CALL WRITR(6,NMSGTX,MSGBUF(1),1,IER)
436   CONTINUE
      CALL CLOSE(6,IER)
C
      P4FLG=1
C
      J=1
      DO 450 I=1,IEND
      IF(SGNSTAT(INUM(I)).GE.1)GO TO 450
      MTLSGN(J)=INUM(I)
      J=J+1
450   CONTINUE
C
      IF(INUM(ICOUNT).EQ.2)MSGBUF(8)=1200
      IF (MSGBUF(8).EQ.1200)
     *   TYPE "[illegible]"
      IF (MSGBUF(8).NE.1200)
     *   WRITE BINARY (10) (MSGBUF(I),I=9,31),MSGBUF(20)
C
      CALL REC (TSK0, MDB)
C
      IF(AMODE.EQ.0)GO TO 90 ;STAND ALONE MODE
      AMODE=0 ;YES
      KCNTO=0
      P4FLG=0
      WRITE (10,55)
55    FORMAT (' ', 'UDCSP IN SIGN STAND-ALONE MODE')
      GO TO 900
C
90    CONTINUE
      DO 120 I=1,BUFSIZ
      IF(XMTBUF(I).EQ.0)GO TO 121
      IF(XMTBUF(I).EQ.RCVBUF(I))GO TO 120 ;XMT AND RCV MESSAGES SAME
      IF(I.EQ.1)GO TO 122
      IF(I.EQ.2)GO TO 123
      GO TO 140
122   WRITE(10,3000) RCVBUF(I)
3000  FORMAT (' ', 'CMS MODEM ADDRESS ERROR :',I5)
      GO TO 150
123   IF(RCVBUF(I).EQ.112400K)GO TO 124 ;BREAK SIGNAL ?
      TYPE 'ERROR IN RETURN MESSAGE-WORD 2' ;NO,ERR IN RETURN MESS
      GO TO 150
```

```
[illegible] IF(RCVBUF(I).[illegible])GO TO 130 ;[illegible] MESSAGE WAS END,OK
       TYPE 'UNEAR SIGNAL IN RETURN MESSAGE'
       GO TO 150
121    IF(RCVBUF(I).EQ.0)GO TO 130
       TYPE 'ECHO-BACK MESSAGE DOES NOT TERMINATE WITH 0'
       GO TO 150
120    CONTINUE
C
130    CONTINUE
       GO TO 900
140    WRITE(10:1000) (RCVBUF(I),I=3,13)
       WRITE(10:2000) (RCVBUF(I),I=15,25)
1000   FORMAT(' ',' ERROR IN RETURN MESSAGE:' / ' ',11A2)
2000   FORMAT(' ',' ',11A2)
150    RCNT=RCNT+1
       IF(RCNT.LT.2)GO TO 900 ;TWO ERRORS IN A ROW ?
       RCNT=0 ;YES:RESET ERR COUNT
C***
C***
C
       DGNSTAT(1)=2 ;ERR FLAG SET
C
C***
C***
       ALMFLG=1 ;FLAG SET TO WRITE TO 'ALMRFC' IN ALM308
       TYPE 'CMS MALFUNCTION'
       GO TO 900
C
900    CONTINUE
       ET1400=0
       RETURN
       END
```

```
      OVERLAY DT15
      SUBROUTINE T15 (ICHAR,ICHCNT,NVAR)
C
      COMMON/GRAPH/MAPNUM,COLOR
C
      INTEGER ICHAR(ICHCNT)
      INTEGER MAPNUM,COLOR
C
C
C
C
C
      IF(NVAR.NE.1)GOTO 1
      IF(ICHAR(2).NE.'  ')GOTO 1
      ICOL=-1
      MAPIND=-1
      IF(ICHAR(1).EQ.'D ')MAPIND=1
      IF(ICHAR(1).EQ.'[illegible] ')MAPIND=2
      IF(ICHAR(1).EQ.'W ')MAPIND=3
      IF(ICHAR(1).EQ.'S ')MAPIND=4
      IF(ICHAR(1).EQ.'R ')MAPIND=5
      IF(ICHAR(1).EQ.'M ')MAPIND=6
      IF(ICHAR(1).EQ.'C ')MAPIND=7
      IF(ICHAR(1).EQ.'N ')MAPIND=8
      IF(ICHAR(1).EQ.'O ')ICOL=0
      IF(ICHAR(1).EQ.'[illegible] ')ICOL=1
      IF(ICOL.EQ.-1)GOTO 1
      COLOR=ICOL
      IF(MAPIND.EQ. -1) GOTO 100
      MAPNUM=MAPIND
      GOTO 100
1     IF(MAPIND.EQ.-1)GOTO 2
      MAPNUM=MAPIND
      GOTO 100
2     TYPE "COMMAND NOT AUTHORIZED"
C
100   RETURN
      END
      OVERLAY OTI64
      SUBROUTINE T164 (INUM,ICHAR,ICOUNT,ICHCNT,FLAG)
      COMMON/WPY/WEATHER(36)
      COMMON/WWDAY/WWB(3),MDY(3)
      COMMON/WWEC/KL300,NWREC,KINC,KSM,NALREC,KDISP
      COMMON/INCA/INCTV(44),INCREC(21),INCFLAG(21),INCSTAT(21)
C
      INTEGER INUM(0:ICOUNT),ICHAR(ICHCNT),FLAG,AVRGE,INCBUF(4)
      INTEGER WWB,MDY,WEATHER
C
C
C
      FLAG=0
      IF(ICHCNT.EQ.2)GOTO 10
1     TYPE "COMMAND NOT AUTHORIZED!"
      GOTO 1000
2     TYPE "DATA ERROR."
      GOTO 1000
10    IF(ICOUNT.LT.2)GOTO 1
      IF(INUM(1).LT.1)GOTO 2
      IF(INUM(2).GE.INUM(1))GOTO 100
      IF(ICHAR(1).NE.'I ')GOTO 2
C
C     'I' COMMAND
C
      IF(ICOUNT.NE.2)GOTO 1
      IF(INUM(2).NE.0)GOTO 2
      IF(INUM(1).GT.22)GOTO 2
      FLAG=1
      GOTO 1000
100   IF(ICHAR(1).NE.'I ')GOTO 200
      IF(INUM(2).GT.22)GOTO 2
      IF(ICOUNT.NE.6)GOTO 110
      IF(INUM(2).EQ.22)GOTO 2
      AVRGE=0
      IF(INUM(1).GT.3)GOTO 103
101   DO 102 I=1,4
      INCBUF(I)=INCTV(I+4)
      INCTV(I+4)=INCTV(I+8)
102   INCTV(I+8)=INCBUF(I)
      IF(AVRGE.EQ.1)GOTO 107
      AVRGE=1
C
C     'I' THRESHOLD MODE. IE.. I M1 M2 N1 N2 N3 N4
C
103   M1=INUM(1)
      M2=INUM(2)
      DO 105 I=M1,M2
      J=4*I-4
      DO 105 K=1,4
105   INCTV(K+J)=INUM(K+2)
      IF(AVRGE.EQ.1)GOTO 101
107   WRITE(10,108)M1,(INUM(I),I=3,6)
108   FORMAT(/1X,'STATION ',I2,' THRESHOLD: ',4(I3,1X))
      IF(M1.NE.M2)WRITE(10,109)M2
109   FORMAT(/1X,'TO STATION ',I2)
      GOTO 1000
C
110   IF(ICOUNT.NE.2)GOTO 2
      IF(INUM(1).LT.INUM(2))GOTO 120
      FLAG=2
      GOTO 1000
120   IF(KINC.GT.1)GOTO 122
      TYPE "NO INCIDENTS"
      GOTO 1000
122   FLAG=3
C
C     HEADING FOR 'I' DISPLAY MODE. IE.. I M1 M2
C
      WRITE(10,125)MDY(3),MDY(1),MDY(2),(WEATHER(I),I=8,16)
125   FORMAT(/1X,'DATE: 19',I2,2('-',I2),' WEATHER: ',9A2,//1X,
     +'DEN FREEWAY INCIDENT SUMMARY',//1X,
     +'STATION  LANE  BEGIN     END     MAX.PATHLINE @  ON/VOL  VERIFIED')
      GOTO 1000
200   IF(ICOUNT.NE.2)GOTO 1
      IF(ICHAR(1).NE.'D ')GOTO 300
C
C     FLAG=5 -> 'D' COMMAND
C
      IF(INUM(2).LE.144)GOTO 210
      IF(INUM(1).GT.30)GOTO 2
      IF(INUM(2).GT.537)GOTO 2
210   FLAG=5
      GOTO 1000
C
```

```
C     FLAG=6 -> 'S' COMMAND
C
300   IF(INUM(2).GT.22)GOTO 2
      IF(ICHAR(1).NE.'S ')GOTO 400
      FLAG=6
      GOTO 1000
C
C     FLAG=7 -> 'V' COMMAND
C
400   IF(ICHAR(1).NE.'V ')GOTO 1
      FLAG=7
C
C
1000  RETURN
      END
      OVERLAY OI16B
      SUBROUTINE TI6B (INUM,ICOUNT,FLAG)
C
C
      COMMON/MRDAY/NMB(3),MDY(3)
      COMMON/NREC/KL308,NMREC,KINC,KSM,NALREC,KDISP
C
      INTEGER INUM(8),ICOUNT),FLAG
      INTEGER NMB,MDY
      INTEGER INCBUF(25)
C
C     'I' VERIFY MODE, IE.. I M N
C
      N=KINC
      CALL OPEN(6,"INCLOG",2,IER,50)
      IF(INUM(2).EQ.0)GOTO 621
610   N=N-1
      IF(N.GT.0)GOTO 620
      TYPE "NO INCIDENT"
      GOTO 1000
620   CALL READR(6,N,INCBUF(1),1,IER)
      IF(INCBUF(4).NE.INUM(2))GOTO 610
621   TYPE " "
      TYPE "STATION LANE BEGIN    END"
      WRITE (10,624)
624   FORMAT (" ", "   --      --  -- -- -- -- -- --")
      IF(INUM(2).EQ.0)GOTO 635
C
C     'I' ADD MODE, IE.. I N 0
C
      IF(INCBUF(9).GT.INCBUF(6))GOTO 630
      WRITE(10,625)(INCBUF(I),I=4,8)
625   FORMAT(/1X,3X,I2,4X,I2,2X,I2,2(':',I2),1X,'-')
      GOTO 634
630   WRITE(10,632)(INCBUF(I),I=4,11)
632   FORMAT(/1X,3X,I2,4X,I2,2X,I2,2(':',I2),1X,I2,2(':',I2))
634   WRITE(10,99)
99    FORMAT(/1X,'CORRECT INCIDENT?: ',Z)
      READ(11,100)IN
100   FORMAT(A1)
      IF(IN.EQ.'N ')GOTO 610
      WRITE (10,624)
635   READ(11,640)(INCBUF(I),I=18,25)
640   FORMAT(3X,A2,4X,A2,1X,6(1X,A2))
      IF(INUM(2).EQ.0)GOTO 643
      IF(INCBUF(19).EQ.'0 ')GOTO 690
      IF(INCBUF(19).EQ.'- ')GOTO 695
C
643   DO 665 I=18,25
      IF(I.NE.23)GOTO 650
      IF(INCBUF(I).NE.'- ')GOTO 651
      INCBUF(I)=20060K
650   IF(I.EQ.19)GOTO 665
651   ILEFT=ISHFT(INCBUF(I),-8)-48
      IF(ILEFT.EQ.-16)ILEFT=0
      IF(ILEFT.LT.0)GOTO 655
      IF(ILEFT.LT.10)GOTO 660
655   TYPE "ILLEGAL DATA"
      GOTO 621
660   IRIGHT=INCBUF(I).AND.377K-48
      IF(IRIGHT.EQ.-16)IRIGHT=0
      IF(IRIGHT.GT.9)GOTO 655
      IF(IRIGHT.LT.0)GOTO 655
      INCBUF(I)=10*ILEFT+IRIGHT
665   CONTINUE
C
      IF(INCBUF(23).EQ.0)GOTO 670
      IEND=60*INCBUF(23)+INCBUF(24)
      ISTART=INCBUF(20)*60+INCBUF(21)
      IF(ISTART.GE.IEND)GOTO 655
      IF(INCBUF(23).GT.23)GOTO 655
      IF(INCBUF(24).GT.59)GOTO 655
      IF(INCBUF(25).GT.59)GOTO 655
670   IF(INCBUF(18).GT.21)GOTO 655
      IF(INCBUF(20).GT.23)GOTO 655
      IF(INCBUF(21).GT.59)GOTO 655
      IF(INCBUF(22).GT.59)GOTO 655
      INCBUF(17)=1   ;TRUE
      IF(INUM(2).NE.0)GOTO 900
C
C     'I' ADD MODE, IE.. I N 0
C
      KINC=N+1
      INCBUF(1)=MDY(3)
```

```
          INCBUF(2)=MDY(1)
          INCBUF(3)=MDY(2)
          DO 680 I=4,16
680       INCBUF(I)=0
          GOTO 900
690       INCBUF(17)=-1  !FALSE
          GOTO 900
695       INCBUF(17)=0  !UNVERIFIED
C
900       CALL WRITR(6,N,INCBUF(1),1,IER)
C
1000      CALL CLOSE(6,IER)
          RETURN
          END
```

```
C
      OVERLAY DT16C
      SUBROUTINE T16C (INUM,ICOUNT,FLAG)
C
      COMMON/MKDAY/HMB(3),MDY(3)
      COMMON/NREC/KL305,NNREC,KINC,KSM,NALRFC,KDISP
C
      INTEGER INUM(0:ICOUNT),FLAG,INCBUF(448),STN
      INTEGER HMB,MDY
C
C
C
      IF(FLAG.NE.3)GOTO 300
C
C     FLAG=3 -> 'I' DISPLAY MODE, IE.. I N1 N2
C
      IND=0
      N1=INUM(1)
      N2=INUM(2)
      CALL OPEN(6,"INCLOG",1,IER,50)
      DO 250 STN=N1,N2
      N=0
207   N=N+1
      IF(N.GE.KINC)GOTO 250
      CALL READR(6,N,INCBUF(1),1,IER)
      IF(INCBUF(4).NE.STN)GOTO 207
      IND=1
      ISTART=INCBUF(6)*60+INCBUF(7)
      IEND=INCBUF(9)*60+INCBUF(10)
      IF(IEND.GT.ISTART)GOTO 208
      WRITE(10,215)(INCBUF(I),I=4,8),INCBUF(13),INCBUF(14)
215   FORMAT(/1X,2X,I2,6X,I2,2X,I2,2(':',I2),3X,' -',10X,I3,9X,I4,6X,Z)
      GOTO 230
208   WRITE(10,235)(INCBUF(I),I=4,11),INCBUF(13),INCBUF(14)
235   FORMAT(/1X,2X,I2,6X,I2,2X,I2,2(':',I2),1X,I2,2(':',I2),5X,I3,9X,I4,6X,Z)
230   IF(INCBUF(17).NE.1)GOTO 240
      TYPE "YES"
      GOTO 207
240   TYPE "NO"
      GOTO 207
250   CONTINUE
      CALL CLOSE(6,IER)
      IF(IND.EQ.0)WRITE(10,255)INUM(1),INUM(2)
255   FORMAT(/1X,'NO INCIDENTS FROM STATION ',I2,' TO STATION ',I2)
      GOTO 1000
C
C     FLAG=7 -> 'V' COMMAND
C
300   N=KSM-1
      IF(N.GT.0)GOTO 302
      TYPE "NO DATA AVAILABLE"
      GOTO 1000
302   N1=INUM(1)
      N2=INUM(2)
      CALL OPEN(6,"DATASM",1,IER,128)
      CALL READR(6,N,INCBUF(1),7,IER)
      CALL CLOSE(6,IER)
      WRITE(10,305)MDY(3),MDY(1),MDY(2),(INCBUF(I),I=1,3)
305   FORMAT(/1X,'19',I2,2('-',I2),3X,I2,2(':',I2),//1X,
     *'5-MINUTE TRUCK AND CAR COUNT',//1X,
     *8X,'STATION    TRUCK     CAR    SUM    TRUCK %' /)
C
      DO 350 STN=N1,N2
      NCAR=0
      NTRUCK=0
      NCON=163-3*STN
      IF(STN.GE.6)NCON=NCON-1
      NT=NCON+260
      HALF1=0
      HALF2=0
      IEND=2
      IF(STN.EQ.6)IEND=3
C
      DO 310 I=0,IEND
      IND2=0
      IND=0
      IF(INCBUF(NCON+I).GE.0)GOTO 306
      HALF1=HALF1+1
      IND=1
306   IF(INCBUF(NT+I).GE.0)GOTO 309
      HALF2=HALF2+1
      IND2=1
309   IF(IND.EQ.0)NCAR=NCAR+INCBUF(NCON+I)
310   IF(IND2.EQ.0)NTRUCK=INCBUF(NT+I)+NTRUCK
C
```

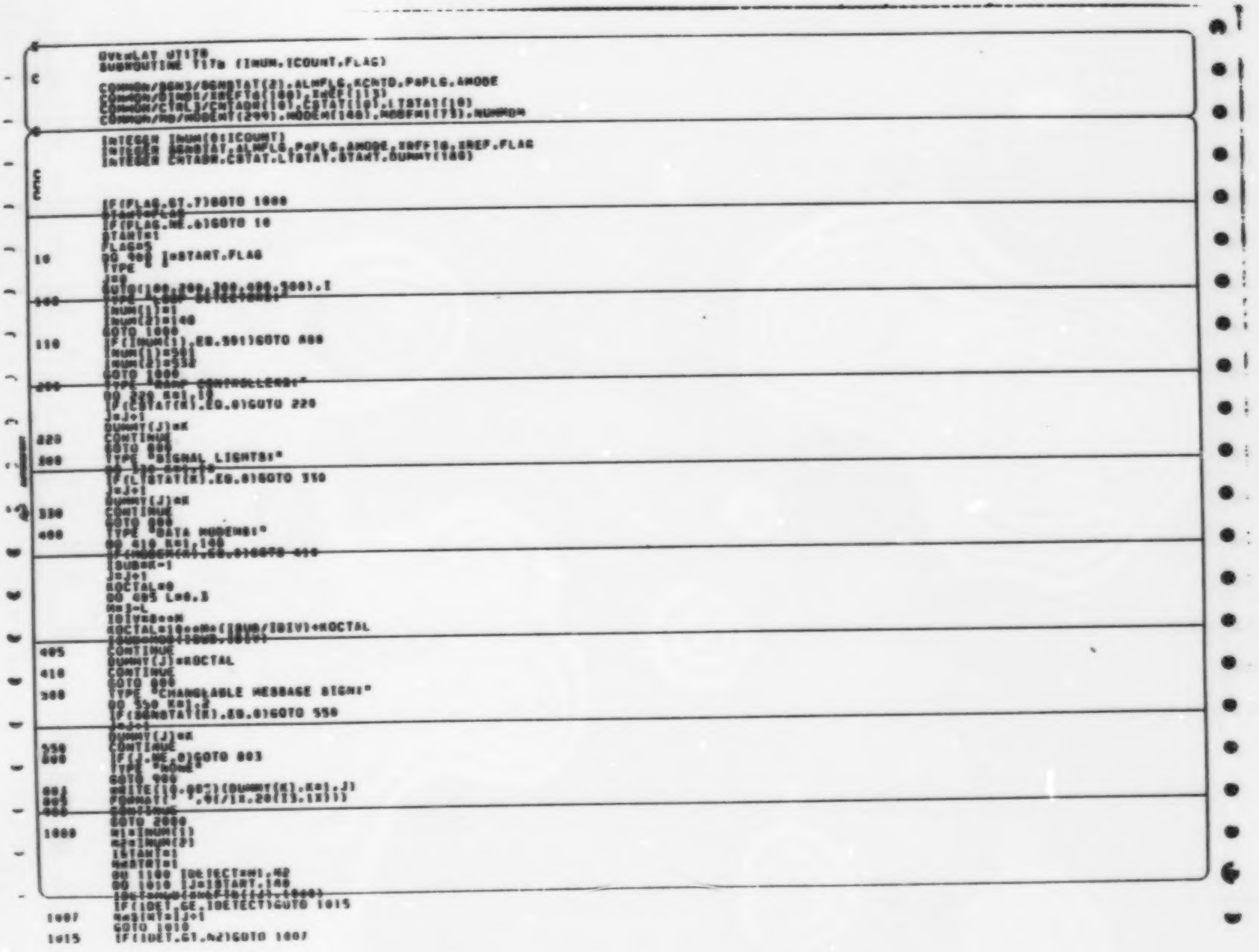
```
      IF(DATA30(NSPD+N).GE.0)GOTO 210
      ISMBUF(N+1)=-1
      HALF=HALF+1
      GOTO 220
210   ISMBUF(N+1)=DATA30(NSPD+N)*36/10
      NAVE=NAVE+ISMBUF(N+1)
220   CONTINUE
C
      IF(HALF.NE.IEND+1)GOTO 224
      NAVE=-1
      GOTO 225
224   NAVE=NAVE/(IEND+1-HALF)
225   IF(I.NE.6)GOTO 222
      WRITE(10,226)I,(ISMBUF(K),K=1,6),NAVE
226   FORMAT(1X,I2,4X,I3,2X,I3,1X,I3,2X,I3,2X,I3)
      GOTO 230
222   WRITE(10,227)I,(ISMBUF(K),K=1,5),NAVE
227   FORMAT(1X,I2,4X,I3,2X,I3,6X,I3,2X,I3,3X,I3)
230   CONTINUE
C
C
1000  RETURN
      END
```

```
C      OVERLAY UT17A
       SUBROUTINE T17A (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,IFLAG)
C
       COMMON/BGN3/BGNSTAT(2),ALMFLG,ACNID,P4FLG,AMODE
       COMMON/CTRL3/CNTADR(10),CSTAT(10),LTSTAT(10)
       COMMON/ALARM/NOBEEP,ALMBEEP
       COMMON/INCH/OCCOMI(21),OCCUMI(21),INCNT(21),INCONT,INCEX
       COMMON/MD/MODEM1(24),MODEM(140),MONPM1(75),NUMMON
C
       INTEGER ICODE(NVAR),CHECK,HOLD(2),INUM(0:ICOUNT),ICHAR(ICHCNT)
       INTEGER BGNSTAT,ALMFLG,P4FLG,AMODE,ALMBEEP
       INTEGER OCCOMI,OCCUMI,CNTADR,CSTAT,LTSTAT
C
C
C
       IFLAG=7
       IF(NVAR.NE.1)GOTO 300
       IF(ICHAR(1).NE.'P ')GOTO 10
       IF(ICHAR(2).NE.'  ')GOTO 1
       IFLAG=6
       GOTO 1000
10     IF(ICHCNT.EQ.3)GOTO 100
1      TYPE "COMMAND NOT AUTHORIZED!"
       GOTO 1000
2      TYPE "DATA ERROR."
       GOTO 1000
100    IF(ICHAR(1).NE.'N ')GOTO 200
       IF(ICHAR(2).NE.'A ')GOTO 120
       NOBEEP=0
       GOTO 1000
120    IF(ICHAR(2).NE.'N ')GOTO 140
       INCONT=0
       ALMBEEP=0
       GOTO 1000
140    IF(ICHAR(2).NE.'D ')GOTO 1
       NOBEEP=1
       GOTO 1000
200    IF(ICHAR(1).NE.'B ')GOTO 1
       IF(ICHAR(2).NE.'E ')GOTO 220
       BGNSTAT(1)=0
       BGNSTAT(2)=0
       GOTO 1000
220    IF(ICHAR(2).NE.'D ')GOTO 1
       BGNSTAT(1)=1
       BGNSTAT(2)=1
       GOTO 1000
300    IF(NVAR.NE.2)GOTO 400
       IF(ICHAR(1).NE.'P ')GOTO 1
       IF(ICHAR(2).NE.'  ')GOTO 1
       IF(ICODE(2).NE.1)GOTO 1
       IF(INUM(1).LT.1)GOTO 2
       IF(INUM(1).GT.5)GOTO 2
       IFLAG=INUM(1)
       GOTO 1000
400    IF(ICHCNT.NE.2)GOTO 1
       IF(ICOUNT.NE.2)GOTO 1
       IF(INUM(1).GT.INUM(2))GOTO 2
       IF(ICHAR(1).EQ.'M ')GOTO 700
       IF(INUM(1).LT.1)GOTO 2
       IF(ICHAR(1).EQ.'K ')GOTO 600
       IF(ICHAR(1).EQ.'D ')GOTO 600
       N1=INUM(1)
       N2=INUM(2)
       IF(ICHAR(1).EQ.'C ')GOTO 410
       IF(ICHAR(1).EQ.'L ')GOTO 410
       GOTO 1
410    IF(INUM(2).GT.10)GOTO 2
       DO 450 I=N1,N2
       IF(ICHAR(1).EQ.'C ')GOTO 420
       LTSTAT(I)=0
       GOTO 450
420    CSTAT(I)=0
450    CONTINUE
       GOTO 1000
600    IF(INUM(2).LE.140)GOTO 650
       IF(INUM(1).LT.501)GOTO 2
       IF(INUM(2).GT.532)GOTO 2
650    IFLAG=8
       IF(ICHAR(1).EQ.'K ')IFLAG=9
       GOTO 1000
700    IF(INUM(2).GT.223)GOTO 2
       DO 740 I=1,2
       HOLD(I)=0
       DO 730 J=1,5
```

```
      LANES=IEND+1
      IF(HALF1.NE.LANES.OR.HALF2.NE.LANES)GOTO 320
      NCAR=-1
      NTRUCK=-1
      ISUM=-1
      IPCNTT=-1
      GOTO 345
320   IF(HALF1.NE.LANES)GOTO 330
      NCAR=-1
      IF(NTRUCK.EQ.0)GOTO 325
      ISUM=NTRUCK
      IPCNTT=100
      GOTO 345
325   ISUM=0
      IPCNTT=0
      GOTO 345
330   IF(HALF2.NE.LANES)GOTO 340
      NTRUCK=-1
      IPCNTT=-1
      ISUM=NCAR
      GOTO 345
340   IPCNTT=0
      ISUM=NTRUCK+NCAR
      IF(ISUM.EQ.0)GOTO 345
      IPCNTT=(NTRUCK*100)/ISUM
345   WRITE(10,346)STN,NTRUCK,NCAR,ISUM,IPCNTT
346   FORMAT(1X,10X,I2,7X,I3,5X,I3,3X,I3,5X,I3)
350   CONTINUE
C
C
1000  RETURN
      END
```

```
C
      OVERLAY OT16D
      SUBROUTINE T16D (INUM,ICOUNT,FLAG)
C
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/DIMUX/XREFTB(180),XREF(113)
      COMMON/NREC/KL30S,NNREC,KINC,K5M,NALREC,KDISP
      COMMON/D30/DATA30(710)
      COMMON/HMDAT/HMS(3),MDY(3)
C
      INTEGER INUM(0:ICOUNT),FLAG,DATA30,ISMBUF(448),HMS,MDY
      INTEGER XREFTB,XREF,STOT,DTOT,CAP
C
C
C
      N1=INUM(1)
      N2=INUM(2)
      IF(FLAG.EQ.6)GOTO 200
C
C     FLAG=5 -> 'D' COMMAND
C
      N=K5M-7
      IF(N.GT.0)GOTO 100
10    TYPE "NO DATA AVAILABLE"
      GOTO 1000
100   CALL OPEN(6,"DATA5M",1,IER,120)
      CALL READR(6,N,ISMBUF(1),7,IER)
      CALL CLOSE(6,IER)
      WRITE(10,105)MDY(3),MDY(1),MDY(2),(ISMBUF(I),I=1,3)
105   FORMAT(/1X,'19',I2,2('-',I2),3X,I2,2(':',I2),//1X,
     +'5-MINUTE DETECTION DATA',//1X,'DETECTOR',10X,'VOLUME',10X,
     +'OCCUPANCY %' /)
      NNSTRT=1
      ISTART=1
C
      DO 150 I=N1,N2
C
C
      DO 140 J=ISTART,180
      IDET=MOD(XREFTB(J),1000)
      IF(IDET.GE.I)GOTO 120
110   NNSTRT=J+1
      GOTO 140
120   IF(IDET.GT.N2)GOTO 110
      DO 122 IPOS=NNSTRT,180
      IDET=MOD(XREFTB(IPOS),1000)
      IF(IDET.EQ.I)GOTO 145
122   CONTINUE
140   CONTINUE
C
145   ISTART=NNSTRT
      IF(IPOS.GT.STOT)GOTO 155
      NVOL=ISMBUF(IPOS+3)
      NOCC=ISMBUF(IPOS+STOT+3)
      GOTO 160
155   NVOL=ISMBUF(IPOS+STOT+3)
      NOCC=ISMBUF(STOT+2*DTOT+IPOS+3)
160   IF(NOCC.LT.1)GOTO 170
      NOCC=(NOCC+35)/75
170   WRITE(10,175)I,NVOL,NOCC
175   FORMAT(1X,3X,I3,13X,I3,15X,I3)
150   CONTINUE
      GOTO 1000
C
C     FLAG=6 -> 'S' COMMAND
C
200   N=KL30S-12
      IF(N.LT.0)GOTO 10
      CALL OPEN(6,"DATA30",1,IER,120)
      CALL READR(6,N,DATA30,12,IER)
      CALL CLOSE(6,IER)
      WRITE(10,215)MDY(3),MDY(1),MDY(2),(DATA30(I),I=1,3)
215   FORMAT(/1X,'19',I2,2('-',I2),3X,I2,2(':',I2),//1X,
     +'30-SECOND MAINLINE SPEED DATA IN KM/H',//1X,
     +8X,'STATION  DL    CL         PL    AVE' /)
C
      DO 230 I=N1,N2
      HALF=0
      IEND=2
      IF(I.EQ.6)IEND=3
      NSPD=3*I-3+1
      IF(I.LT.6)NSPD=NSPD+1
      NAVE=0
C
      DO 220 M=0,IEND
```

```
        IDIV=10**(3-J)
        CHECK=MOD(INUM(I),10*IDIV)/IDIV
        IF(CHECK.GT.7)GOTO 2
730     HOLD(I)=CHECK*8**(3-J)+HOLD(I)
740     HOLD(I)=HOLD(I)+1
        N1=HOLD(1)
        N2=HOLD(2)
        DO 750 I=N1,N2
750     NUMBER(I)=0
C
1000    RETURN
        END
```



```
      OVERLAY UTITB
      SUBROUTINE TITB (INUM,ICOUNT,FLAG)
C
      COMMON/BGM3/BGMSTAT(2),ALMFLG,KCNTD,P0FLG,AMODE
      COMMON/DIMDS/INFFTB(100),IREF(113)
      COMMON/CTRL3/CNTADR(10),CSTAT(10),LTSTAT(10)
      COMMON/MD/MODEMT(299),MODEM(140),MODFMI(73),NUMMDM
      INTEGER INUM(0:ICOUNT)
      INTEGER BGMSTAT,ALMFLG,P0FLG,AMODE,INFFTB,IREF,FLAG
      INTEGER CNTADR,CSTAT,LTSTAT,START,DUMMY(140)
C
C
C
      IF(FLAG.GT.7)GOTO 1000
      START=FLAG
      IF(FLAG.NE.6)GOTO 10
      START=1
      FLAG=5
10    DO 900 I=START,FLAG
      TYPE " "
      J=0
      GOTO(100,200,300,400,500),I
100   TYPE "LOOP DETECTORS:"
      INUM(1)=1
      INUM(2)=140
      GOTO 1000
110   IF(INUM(1).EQ.501)GOTO 400
      INUM(1)=501
      INUM(2)=532
      GOTO 1000
200   TYPE "RAMP CONTROLLERS:"
      DO 220 K=1,10
      IF(CSTAT(K).EQ.0)GOTO 220
      J=J+1
      DUMMY(J)=K
220   CONTINUE
      GOTO 800
300   TYPE "SIGNAL LIGHTS:"
      DO 330 K=1,10
      IF(LTSTAT(K).EQ.0)GOTO 330
      J=J+1
      DUMMY(J)=K
330   CONTINUE
      GOTO 800
400   TYPE "DATA MODEMS:"
      DO 410 K=1,140
      IF(MODEM(K).EQ.0)GOTO 410
      ISUB=K-1
      J=J+1
      KOCTAL=0
      DO 405 L=0,3
      M=3-L
      IDIV=8**M
      KOCTAL=10**M*(ISUB/IDIV)+KOCTAL
      ISUB=MOD(ISUB,IDIV)
405   CONTINUE
      DUMMY(J)=KOCTAL
410   CONTINUE
      GOTO 800
500   TYPE "CHANGEABLE MESSAGE SIGNS:"
      DO 550 K=1,2
      IF(BGMSTAT(K).EQ.0)GOTO 550
      J=J+1
      DUMMY(J)=K
550   CONTINUE
800   IF(J.NE.0)GOTO 803
      TYPE "NONE"
      GOTO 900
803   WRITE(10,805)(DUMMY(K),K=1,J)
805   FORMAT(" ",6(/1X,20(I3,1X)))
900   CONTINUE
      GOTO 2000
1000  N1=INUM(1)
      N2=INUM(2)
      ISTART=1
      NSTRT=1
      DO 1100 IDETECT=N1,N2
      DO 1010 IJ=ISTART,140
      IDET=[illegible](INFFTB(IJ)-1000)
      IF(IDET.GE.IDETECT)GOTO 1015
1007  NSINT=IJ+1
      GOTO 1010
1015  IF(IDET.GT.N2)GOTO 1007
```

```
      DO 1050 IPOS=NSTRT,100
      IDET=MOD(XREFTB(IPOS),1000)
      IF(IDET.EQ.IDETECT)GOTO 1020
1050  CONTINUE
1010  CONTINUE
1020  ISTART=NSTRT
      IF(FLAG.GT.6)GOTO 1030
      IF(XREFTB(IPOS).LT.1000)GOTO 1100
      J=J+1
      DUMMY(J)=IDETECT
      GOTO 1100
1030  XREFTB(IPOS)=IDETECT
      IF(FLAG.EQ.9)XREFTB(IPOS)=XREFTB(IPOS)+1000
1100  CONTINUE
      IF(FLAG.LT.6)GOTO 110
C
2000  RETURN
      END
      OVERLAY OT18A
      SUBROUTINE T18A (ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,FLAG)
C
C
      COMMON/COMM/DETECT(545),NDETRC,NDKEY
      COMMON/NREC/NL300,NAREC,KINC,KSM,NALRFC,KDISP
      COMMON/NRDAY/HMS(3),MDY(3)
      INTEGER ICODE(NVAR),INUM(2:ICOUNT),ICHAR(ICHCNT)
      INTEGER DETECT, NDETRC, NDKEY
      INTEGER HMS,MDY
      INTEGER FLAG, OUTBUF(545)
C
      LOGICAL ITEST
C
C     FLAG=0 : COMMAND COMPLETED
C     FLAG=1 : COMMAND=A
C     FLAG=2 : COMMAND=T
C     FLAG=3 : COMMAND=S
C     FLAG=4 : COMMAND=DI
C     FLAG=5 : COMMAND=DM
C     FLAG=6 : COMMAND=P
C     FLAG=7 : COMMAND=M
C     FLAG=8 : COMMAND=[illegible]
C     FLAG=9 : COMMAND=Z
C
C
C
      FLAG=0
      IF(NVAR.NE.1)GO TO 900
      IF((ICHCNT.LT.3))GO TO 300
      IF(ICHCNT.NE.2)GO TO 1
      IF((ICHAR(1).EQ.'D ').AND.ICHAR(2).EQ.'I ')GO TO 100
      IF((ICHAR(1).EQ.'D ').AND.ICHAR(2).EQ.'M ')GO TO 100
1     TYPE "COMMAND NOT AUTHORIZED!"
      GO TO 2000
2     TYPE "DATA ERROR."
      GO TO 2000
C
100   FLAG=ISHFT(ICHAR(1),-8).AND.7
      GO TO 2000
C
C
300   IF((ICHAR(1).EQ.'D ')GO TO 400
      IF((ICHAR(1).EQ.'T ')GO TO 500
      IF((ICHAR(1).EQ.'M ')GO TO 600
      IF((ICHAR(1).EQ.'[illegible] ')GO TO 700
      IF((ICHAR(1).EQ.'Y ')GO TO 800
      IF((ICHAR(1).EQ.'P ')GO TO 100
      IF((ICHAR(1).NE.'Z ')GO TO 1
C
C
C     'Z' COMMAND
C
C
      FLAG=9
      GO TO 2000
C
C
C     'D' COMMAND
C
C
400   CONTINUE
C
C     WAIT TILL NEXT MINUTE
C
      CALL TIME (HMS(1), IER)
      HMS(3) = 60 - HMS(3)
      CALL WAIT (HMS(3), 2, IER)
C
C     PRINT DATE AND STARTING TIME
C
      CALL TIME (HMS(1), IER)
      WRITE (12, 410) (MDY(I), I=1,3), (HMS(I), I=1,3)
410   FORMAT ('1', ' 1 MIN. OF DETECTOR DATA', 10X,
     *          'DATE = ', I2, '/', I2, '/', I2, 10X,
     *          'STARTING TIME = ', I2, ':', I2, ':', I2 /)
C
C     CREATE AND OPEN 'DET60S' FILE
C
      CALL CFILW ('DET60S', 2, IER)
      CALL OPEN (6, 'DET60S', 3, IER, 1090)
C
```

```
C
C     WAIT TILL 60 SEC. HAVE ELAPSED IN TASK 'COMCOM'
C
      NOLINC = 1
      CALL REC (MOKEY, MSGOTN)
      NOLINC = 0
C
C     PRINT 'DETA08' RECORDS
C
      CALL CLOSE (6, IER)
      CALL OPEN (6, 'DETA08', 1, IER, 1090)
      DO 450 I = 1, 64
         CALL READR (6, I, OUTBUF(1), 1, IER)
         WRITE (10, 470) I, (OUTBUF(J), J=1,545)
470      FORMAT (1H , 'RECORD # ', I4, ' IS' / (1X, 15I6) )
450   CONTINUE
C
C     CLOSE AND DELETE 'DETA08' FILE
C
      CALL CLOSE (6, IER)
      CALL DFILW ('DETA08', IER)
      GO TO 2000
C
C
C     'T' COMMAND
C
500   CONTINUE
      CALL TIME (HMS(1), IER)
      WRITE (10, 505) MDY(3),MDY(1),MDY(2),
     *   HMS(1),HMS(2),HMS(3)
505   FORMAT (1H , ' ', I4,'.',I2,2('-',I2),3X,I2,2(':',I2))
      GO TO 2000
C
C
C     'N' COMMAND
C
C
600   CONTINUE
      FLAG=0
      GO TO 2000
C
C
C     'Y' COMMAND
C
C
800   CONTINUE
      FLAG=2
      GO TO 2000
C
C
900   IF(ICMCNT.NE.3)GO TO 1
      IF(ICOUNT.NE.3)GO TO 1000
      IF(INUM(1).GT.INUM(2))GO TO 2
      IF(INUM(1).LT.1)GO TO 2
      IF(ICHAR(1).NE.'A ')GO TO 910
      IF(INUM(2).GT.10)GO TO 2
      GO TO 100
910   IF(ICHAR(1).NE.'B ')GO TO 1
      IF(INUM(2).GT.3)GO TO 2
      GO TO 100
C
C
1000  IF(NVAR.NE.2)GO TO 1
      IF(ICOUNT.NE.1)GO TO 1
      [illegible] GO TO 1
      TYPE "COMMAND NOT YET IMPLEMENTED"
C
C
2000  RETURN
      END
      OVERLAY UT108
      SUBROUTINE T108 (INUM,ICOUNT,FLAG)
C
C
      COMMON/CTRL3/CNTAGN(10),CSTAT(10),LTSTAT(10)
      COMMON/CTRL6/COMDOP(4),CNTON(10),NBRAMP
      COMMON/SGM1/INTBUF(45),RCVBUF(45),SGNADR(2)
      [illegible]
      COMMON/RANGE/UPRNGE(13,2),DNRNGE(67,3)
      COMMON/WPV/WEATHER(36)
      COMMON/NREC/RL303,NNREC,KINC,KSM,NALNEC,KDISP
C
      INTEGER INUM(0:ICOUNT),FLAG,WEATHER
      INTEGER COMDOP,CNTON,NBRAMP
      INTEGER CNTAGN,CSTAT,LTSTAT
      INTEGER [illegible]
      INTEGER UPRNGE,DNRNGE
      INTEGER OUTBUF(545)
C
C
C
C
      GO TO (500,300,600,100,200,1000,400,800,900), FLAG
C
C
C     'DI' COMMAND
C
C
100   CALL DDIS
      TYPE "CTRL A INTERRUPT DISABLED"
      GO TO 1000
C
C
C     'EN' COMMAND
C
C
200   CALL DEBL
      TYPE "CTRL A INTERRUPT ENABLED"
      GO TO 1000
C
C
C     'T' COMMAND
C
C
300   CONTINUE
      DO 310 I=1,26
310   OUTBUF(I)=UPRNGE(I)
      DO 320 I=1,201
320   OUTBUF(I+26)=DNRNGE(I)
      WRITE (10,330) (OUTBUF(I), I=1,227)
330   FORMAT (' ','UPRNGE(13,2)',2(/1X,7(/1X,13I5)/1X,6I5)
     *  /(1X,'DNRNGE(67,3)',3(/1X,4(/1X,15I5)/1X,7I5))
      GO TO 1000
C
C
C     'W' COMMAND
C
C
400   CALL OPEN(6,"WEATHER",3,IER,72)
      WRITE(10,403)
      READ(11,404)(WEATHER(I),I=1,16)
      WRITE(10,405)
      READ(11,404)(WEATHER(I),I=17,26)
      WRITE(10,406)
      READ(11,404)(WEATHER(I),I=27,36)
403   FORMAT(' ','WEATHER     ',Z)
404   FORMAT([illegible])
405   FORMAT(' ','COMMENT     ',Z)
406   FORMAT(' ','VISIBILITY  ',Z)
      NNREC=1
      CALL WRITR(6,0,WEATHER,1,IER)
      CALL CLOSE(6,IER)
      GO TO 1000
C
C
C     'A' COMMAND
C
C
500   N1=INUM(1)
      N2=INUM(2)
      DO 510 I=N1,N2
510   CNTAGN(I)=INUM(I)
      WRITE(10,515)CNTAGN
515   FORMAT(' ',' CNTAGN',10(1X,I6))
```

```
      GO TO 1000
C
C
C     'S' COMMAND
C
C
600   N1=INUM(1)
      N2=INUM(2)
      DO 610 I=N1,N2
610   SENSOR(I)=INUM(3)
      TYPE "IN SUBR 'TIBB'. SENSOR =",SENSOR
      GO TO 1000
C
C
C     'M' COMMAND
C
C
800   CONTINUE
      DO 810 I=1,15
810   OUTBUF(I)=[illegible](I)
      WRITE (10,830) (OUTBUF(I), I=1,15)
830   FORMAT (' ', [illegible] /15,10(/15,15I5))
      GO TO 1000
C
C
C     'I' COMMAND
C
C
900   CONTINUE
      DO 930 I=1,8
      DO 910 J=1,16
      K=16-J
      OUTBUF(J)=ITEST(COMBOP(I),K) .AND. 1
910   CONTINUE
      WRITE (10,920) I, (OUTBUF(L), L=1,16)
920   FORMAT (' ', 'COMBOP(',I1,') = ',16I1)
930   CONTINUE
      GO TO 1000
C
C
1000  RETURN
      END
      OVERLAY OT10C
      SUBROUTINE T10C
C
      COMMON/BGK2/NMOHDB,NUFSIZ,NMBGTX,NMBGRD,MLTBGN(2)
      COMMON/NREC/XL300,NWREC,KINC,KSM,NALREC,KDISP
      COMMON/WNDAT/NWS(3),NDV(3)
      COMMON/NUFIL/RDRCD,NSOVR(21)
      INTEGER RECFM(11)
      INTEGER NWS,NDV,NUFSIZ
      INTEGER RDRCD,NSOVR
C
C
C
      CALL PRI(0)
      CALL RMDCK                    ;REMOVE USER CLOCK
C
      CALL DESTR (10, IER)          ;DESTROY TASK 'COMCOM'
      CALL DESTR (20, IER)          ;DESTROY TASK 'ALMBGN'
      CALL DESTR (30, IER)          ;DESTROY TASK 'TRANSF'
      CALL DESTR (40, IER)          ;DESTROY TASK 'GMAP'
C
C
      NMBGTX=NMBGTX+1
      RDRCD=RDRCD-1
      WRITE(10,135)XL300,NWREC,KINC,KSM,NALREC,KDISP,NMBGTX,RDRCD
135   FORMAT(/1X,"FILE",5X,"RCDS"//
     +    "    DATA10  : ",I5/
     +    "    WEATHER : ",I5/
     +    "    INCLOG  : ",I5/
     +    "    DATASM  : ",I5/
     +    "    ALMREC  : ",I5/
     +    "    DEMOTR  : ",I5/
     +    "    INITAB  : ",I5/
     +    "    MESDEV  : ",I5//)
C
C
      RECFM(1)=53505K
      RECFM(2)=40537K
      RECFM(3)=[illegible]
      RECFM(4)=51040K
      RECFM(5)=NDV(3)
      RECFM(6)=NDV(1)
      RECFM(7)=NDV(2)
      RECFM(8)=72
      RECFM(9)=1
      RECFM(10)=36
      RECFM(11)=NWREC
      CALL OPEN(6,"WEATHER",3,IER,72)
      CALL WRITR(6,0,RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=44510K
      RECFM(2)=41315K
      RECFM(3)=[illegible]
      RECFM(4)=25040K
      RECFM(8)=50
      RECFM(10)=25
      RECFM(11)=KINC
      CALL OPEN(6,"INCLOG",3,IER,50)
      CALL WRITR(6,0,RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=40514K
      RECFM(2)=44531K
      RECFM(3)=42503K
      RECFM(8)=20
      RECFM(10)=16
      RECFM(11)=NALREC
      CALL OPEN(6,"ALMREC",3,IER,20)
      CALL WRITR(6,0,RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=54115K
      RECFM(2)=52124K
      RECFM(3)=40502K
      RECFM(8)=102
      RECFM(10)=51
      RECFM(11)=NMBGTX
      CALL OPEN(6,"INITAB",3,IER,102)
      CALL WRITR(6,0,RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=[illegible]
      RECFM(2)=[illegible]
      RECFM(3)=[illegible]
      RECFM(8)=52
      RECFM(10)=[illegible]
      RECFM(11)=[illegible]
      CALL OPEN(6,"[illegible]",3,IER,32)
      CALL WRITE(6,[illegible],RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=[illegible]
      RECFM(2)=[illegible]
      RECFM(3)=[illegible]
      RECFM(8)=[illegible]
      RECFM(9)=[illegible]
      RECFM(10)=[illegible]
      RECFM(11)=[illegible]
      CALL OPEN(6,"[illegible]",3,IER,128)
      CALL WRITE(6,[illegible],RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=[illegible]
      RECFM(2)=[illegible]
      RECFM(3)=[illegible]
      RECFM(8)=[illegible]
      RECFM(10)=[illegible]
      RECFM(11)=[illegible]
      CALL OPEN(6,"[illegible]",3,IER,128)
      CALL WRITE(6,[illegible],RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      RECFM(1)=[illegible]
      RECFM(2)=[illegible]
      RECFM(3)=[illegible]
      RECFM(8)=[illegible]
      RECFM(10)=[illegible]
      RECFM(11)=[illegible]
      CALL OPEN(6,"[illegible]",3,IER,128)
      CALL WRITE(6,[illegible],RECFM(1),1,IER)
      CALL CLOSE(6,IER)
C
C
      TYPE "< [illegible] SYSTEM [illegible] [illegible] >"
C
      RETURN
      END
      SUBROUTINE TOPCOM
C
C     THIS ROUTINE PROVIDES THE OPERATOR COMMANDS.
C
      EXTERNAL DIMPLOIT,OT124,OT120,OT13,
     *  [illegible]
     *  OT16A,OT16B,OT16C,OT16D,OT17A,OT17B,
     *  OT18A,OT18B,OT18C
C
      COMMON/BLOK1/FT140
C
      INTEGER COMAND,COM1,COM2,ICODE(30),INUM(0:30),ICHAR(60)
      INTEGER FT140
C
C
C
      CALL OVOPN (13, 'GEN.OL', IER)
      INUM(0)=0
      ICOUNT=30
      ICHCNT=60
      NVAR=30
C
3     CONTINUE
      WRITE(10,103)
103   FORMAT(' ',' GEN*')
      GO TO 4
C
4     CONTINUE
      WRITE(10,101)
101   [illegible]
      READ(11,102)COM1,COM2 ;COMMAND GROUP
102   FORMAT(2A2)
C
      IF(COM2.EQ.'  ')GO TO 30
      IF(COM1.EQ.'CM'.AND.COM2.EQ.'D ')GO TO 10
      IF(COM1.EQ.'TM'.AND.COM2.EQ.'D ')GO TO 10
      IF(COM1.NE.'SC'.OR.COM2.NE.'W ')GO TO 1
10    [illegible]
      GO TO 1000
C
30    IF(COM1.EQ.'MC')GO TO 90
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      IF(COM1.EQ.' ')GO TO 1
1     TYPE "INVALID COMMAND GROUP!"
      GO TO 4
C
C
1000  COMAND=(ISHFT(COM1,-8).AND.000016K)/2+1
      CALL FOVLD(11,DIMPLOIT,0,IER)
      CALL IMPLOIT(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,COM1,COM2)
      CALL FOVRL(DIMPLOIT,IER)
      IF(ICODE(1).EQ.2)GO TO 85
      TYPE "COMMAND NOT AUTHORIZED!"
      GO TO 1000
C
85    IF(ICHAR(1).NE.'E ')GO TO 90 ; EXIT COMMAND GROUP
      IF(ICHAR(2).NE.' ')GO TO 90
      IF(NVAR.EQ.1)GO TO 3
C
90    GO TO(200,100,300,400,600,500,700),COMAND
C
C
C     'MC' COMMAND GROUP
C
C
100   CALL FOVLD(13,OT124,0,IER)
      CALL T124(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,IFLAG)
      CALL FOVRL(OT124,IER)
      IF(IFLAG.EQ.1)GO TO 1000
      CALL FOVLD(13,OT120,0,IER)
      CALL T120(INUM,ICOUNT,ICHCNT)
      CALL FOVRL(OT120,IER)
      GO TO 1000
C
C
C     'GD' COMMAND GROUP
C
C
200   CALL FOVLD(13,OT13,0,IER)
      CALL T13(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
      CALL FOVRL(OT13,IER)
      GO TO 1000
C
C
C     'CMS' COMMAND GROUP
C
C
300   CALL FOVLD(13,OT14A,0,IER)
      CALL T14A(ICODE,ICHAR,NVAR,ICHCNT,IFLAG)
      CALL FOVRL(OT14A,IER)
      GO TO(1000,310,320),IFLAG
310   CALL FOVLD(13,OT14B,0,IER)
      CALL T14B(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
      CALL FOVRL(OT14B,IER)
      GO TO 1000
320   CALL FOVLD(13,OT14C,0,IER)
      CALL T14C(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT)
      CALL FOVRL(OT14C,IER)
      CALL FOVLD(13,OT14D,0,IER)
      CALL T14D (INUM,ICOUNT)
      CALL FOVRL(OT14D,IER)
      GO TO 1000
C
C
C     'S' COMMAND GROUP
C
C
400   CALL FOVLD(13,OT15,0,IER)
      CALL T15(ICHAR,ICHCNT,NVAR)
      CALL FOVRL(OT15,IER)
      GO TO 1000
C
C
C     'INP' COMMAND GROUP
C
C
500   CALL FOVLD(13,OT16A,0,IER)
      CALL T16A(INUM,ICHAR,ICOUNT,ICHCNT,IFLAG)
      CALL FOVRL(OT16A,IER)
      GO TO(510,510,520,1000,530,530,520),IFLAG
510   CALL FOVLD(13,OT16B,0,IER)
      CALL T16B(INUM,ICOUNT,IFLAG)
      CALL FOVRL(OT16B,IER)
      GO TO 1000
520   CALL FOVLD(13,OT16C,0,IER)
      CALL T16C(INUM,ICOUNT,IFLAG)
      CALL FOVRL(OT16C,IER)
      GO TO 1000
530   CALL FOVLD(13,OT16D,0,IER)
      CALL T16D(INUM,ICOUNT,IFLAG)
      CALL FOVRL(OT16D,IER)
      GO TO 1000
C
C
C     'N' COMMAND GROUP
C
C
600   CALL FOVLD(13,OT17A,0,IER)
      CALL T17A(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,IFLAG)
      CALL FOVRL(OT17A,IER)
      IF(IFLAG.EQ.7)GO TO 1000
      CALL FOVLD(13,OT17B,0,IER)
      ICOUNT=2
      CALL T17B(INUM,ICOUNT,IFLAG)
      CALL FOVRL(OT17B,IER)
      GO TO 1000
C
C
C     'JCW' COMMAND GROUP
C
C
700   CALL FOVLD(13,OT18A,0,IER)
      CALL T18A(ICODE,INUM,ICHAR,NVAR,ICOUNT,ICHCNT,IFLAG)
      CALL FOVRL(OT18A,IER)
      IF(IFLAG.EQ.0)GO TO 1000
      IF(IFLAG.EQ.6)GO TO 710
      CALL FOVLD(13,OT18B,0,IER)
      CALL T18B(INUM,ICOUNT,IFLAG)
      CALL FOVRL(OT18B,IER)
      GO TO 1000
C
710   CALL FOVLD(13,OT18C,0,IER)
      CALL T18C
      CALL FOVRL(OT18C,IER)
C
      CALL CLOSE(13,IER)
```

END

```
      TASK THANSF
C
C
C
C     A THIRTY-SECOND PERIODIC FREEWAY TRAFFIC SURVEILLANCF
C       AND CONTROL TASK, INCLUDING INCIDENT DETECTION
C       AND RAMP CONTROL.
C
      EXTERNAL OMOOD30,OUPD308,OINCDET,OPERCHK
      EXTERNAL ODATSTR,OCLDU,OVTLU,ORCTIME,OQUOCC
      EXTERNAL OALM308,OALM5M,ODIS1,ODY81,ODIS2,OAMODM
C
      COMMUN/TC/IMIN10
      COMMON/TCNT/COUNT,KCNTSM,IMINS,KCOUNT
      COMMON/ALARM/NOBEEP,ALMBEEP
      COMMON/INCD/OCCOUT(21),OCCOUT(21),INCNT(21),INCONT,INCEX
      COMMON/TSKSYN/TSK0,TSK1,TSK2,TSK3
      COMMON/MKDAY/HMS(3),MDY(3)
      COMMON/SCREEN/GCHN,PFILE,RGEN,RWTEMP
C
      INTEGER IMIN10
      INTEGER COUNT,KCNTSM,IMINS,KCUUNT
      INTEGER NOBEEP,ALMBEEP
      INTEGER OCCOUT,OCCOUT,INCNT,INCONT,INCEX
      INTEGER TSK0,TSK1,TSK2,TSK3
      INTEGER HMS,MDY
      INTEGER GCHN,PFILE,RGEN,RWTEMP
C
900   CONTINUE
      CALL REC(TSK1,MD1)
C
      CALL TIME (HMS(1),IER)
X     WRITE (12, 1111) (HMS(I), I=1,3)
X1111 FORMAT (' TRANSF TIME =', 3I5)
      CALL DATE (MDY(1),IER)
C
      CALL FOVLD(1,OMOOD30,0,IER)
      CALL MOOD30
      CALL FOVRL(OMOOD30,IER)
C
      CALL FOVLD(1,OUPD308,0,IER)
      CALL UPD308
      CALL FOVRL(OUPD308,IER)
C
      IF(INCEX.EQ.0)GO TO 100
C
      CALL FOVLD(1,OINCDET,0,IER)
      CALL INCDET
      CALL FOVRL(OINCDET,IER)
C
      CALL FOVLD(1,OPERCHK,0,IER)
      CALL PERCHK
      CALL FOVRL(OPERCHK,IER)
C
      CALL FOVLD(1,ODATSTR,0,IER)
      CALL DATSTR
      CALL FOVRL(ODATSTR,IER)
C
      CALL FOVLD(1,OCLDU,0,IER)
      CALL CLDU
      CALL FOVRL(OCLDU,IER)
C
      CALL FOVLD(1,OVTLU,0,IER)
      CALL VTLU
      CALL FOVRL(OVTLU,IER)
C
100   INCEX=1
C
      CALL FOVLD(1,ORCTIME,0,IER)
      CALL RCTIME
      CALL FOVRL(ORCTIME,IER)
C
      CALL FOVLD(1,OQUOCC,0,IER)
      CALL QUOCC
      CALL FOVRL(OQUOCC,IER)
C
      CALL FOVLD(1,OALM308,0,IER)
      CALL ALM308
      CALL FOVRL(OALM308,IER)
C
      IF(IMINS.EQ.0)GO TO 900
      IMINS=0 ;15 MINS ALM..RESET COUNT
C
      CALL FOVLD(1,OALM5M,0,IER)
      CALL ALMSM
      CALL FOVRL(UALMSM,IER)
C
      IMIN10=IMIN10+1
      IF(IMIN10.NE.2)GO TO 600
      IMIN10=0
C
      CALL FOVLD(1,OAMODM,0,IER)
      CALL AMODM
      CALL FOVRL(UAMODM,IER)
C
600   CALL FOVLD(1,ODIS1,0,IER)
      CALL DIS1
      CALL FOVRL(ODIS1,IER)
C
      CALL FOVLD(1,ODYS1,0,IER)
      CALL DYS1
      CALL FOVRL(ODYS1,IER)
C
      IGO=MOD(KCOUNT,2)
      IF(IGO.EQ.1)GO TO 850           ;NOT END OF MIN YET
C
C
      CALL FOVLD(1,ODIS2,0,IER)
      CALL DIS2
      CALL FOVRL(ODIS2,IER)
C
      CALL XMT (TSK2,1,$1000)
850   IF(NOBEEP.LE.0.AND.ALMBEEP.GT.0)TYPE '<007>';BEEP AT TTY
C     IF (NOBEEP.LE. 0.AND.INCONT.GT. 0) WRITE(GCHN) "<007>"
      GO TO 900
C
1000  TYPE "IN TASK 'TRANSF', XMT ERROR TO TASK 'GMAP'; TSKP =", TSKP
      CALL STTSK (40, ISTGMA, IER)
      TYPE "STTSK IER =", IER, ", GMAP STATUS =", ISTGMA
C
      END
      OVERLAY SUPD30S
      SUBROUTINE UPD30S
C
C DATE: 80-1-10
C    THIS TASK IS EXECUTED ONCE EVERY 30 SECONDS IMMEDIATELY AFTER
C       TASK TRANSF. IT CHECKS DETECTOR FOR MALFUNCTION. IT TAKES ONE
C       OF THE FOLLOWING ACTIONS DEPENDING ON THE DETECTOR STATUS.
C
C    IN PAIRED LOOP DETECTORS:
C    CASE A: IF UPSTREAM DETECTOR NOT O.K. AND DOWNSTREAM DETECTOR NOT O.K.
C            SET CORRESPONDING VOLUME, OCCUPANCY, SPEED AND VEHICLE TYPE
C            TO -1.
C    CASE B: IF UPSTREAM DETECTOR IS O.K. BUT NOT DOWNSTREAM THEN SET
C            CORRESPONDING SPEED, AND VEHICLE TYPE TO -1. BUT USE UPSTREAM
C            VOLUME & OCCUPANCY AS CORRESPONDING DOWNSTREAM VALUES.
C    CASE C: IF UPSTREAM DETECTOR IS NOT O.K. BUT DOWNSTREAM IS O.K. THEN
C            SET CORRESPONDING SPEED AND VEHICLE TYPE TO -1.
C    CASE D: IF UPSTREAM AND DOWNSTREAM DETECTORS ARE O.K. MAKE NO
C            CHANGES
C
C    THIS TASK ALSO CONSOLIDATES LANE INTO STATION DATA. THE STATION
C    DATA IS SET TO -1 IFF ALL THE LANE DATA ARE -1; OTHERWISE, THE
C    REMAINING 'GOOD' LANE DATA ARE USED TO DERIVE STATION DATA.
C
C    CSOCC: LATEST 30 SEC OCC
C    PSOCC: NEXT TO LATEST 30 SEC OCC
C    SOCC1M: LATEST 1 MIN OCC
C    SVOL1M: LATEST 1 MIN VOL
C    CSVOL: LATEST 30 SEC VOL
C    PSVOL: NEXT TO THE LATEST 30 SEC VOL
C    SSPD: LATEST 30 SEC SMOOTHED SPEED
C    LNSTS: NUMBER OF LANES IN A STATION GIVING VOL OCC SSPEED DATA
C
      DIMENSION DATASM(668)
      DIMENSION SDVOL(46),SDOCC(46)
      DIMENSION DNVOL(67),UPVOL(67),DNOCC(67),UPOCC(67),LNSPD(67),
      DIMENSION LVCNT(67)
      DIMENSION CSOCC(22),PSOCC(22),SOCC1M(22),SVOL1M(22),CSVOL(22),
     *PSVOL(22),SSPD(22),LNSTS(22)
      COMMON/DD2/DWAIT(10),QLNTH(10)
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),MANCYC(10)
      COMMON/MRDAT/HMS(3),MDY(3)
      COMMON/GEOM/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/SECTL/DIST(20)
      COMMON/TTS/TT(20)
      [illegible]
      COMMON/DIMDX/XREFTB(100),XREF(113)
      COMMON/TCNT/COUNT,KCNTSM,IMINS,KCOUNT
      COMMON/MREC/KL30S,NNREC,KINC,KSM,MALREC,KDISP
C
      INTEGER DATASM
      INTEGER DWAIT,QLNTH
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER CYCOVR,MANCYC
      INTEGER SDVOL,SDOCC
      INTEGER DNVOL,UPVOL,DNOCC,UPOCC,LNSPD,LVCNT
      INTEGER CSOCC,PSOCC,SOCC1M,SVOL1M,CSVOL,PSVOL,SSPD,LNSTS
      INTEGER HMS,MDY
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER DIST
      INTEGER TT
      INTEGER DATA30
      INTEGER XREFTB,XREF
      INTEGER COUNT,KCNTSM,IMINS,KCOUNT
      INTEGER KL30S,NNREC,KINC,KSM,MALREC,KDISP
      INTEGER DSTAT,DNSTAT,UPSTAT
      INTEGER NPOS,VOLSUM,OCCSUM,SPDSUM,NSP
      INTEGER I,K,J,I1,I2,IK,NLANE,NSTNM2,TSPD
C
      EQUIVALENCE (SDVOL(1),DATA30(49)),(SDOCC(1),DATA30(95)),
     *(DNVOL(1),DATA30(141)),(UPVOL(1),DATA30(208)),
     *(DNOCC(1),DATA30(275)),(UPOCC(1),DATA30(342)),
     *(LNSPD(1),DATA30(409)),(LVCNT(1),DATA30(476))
      EQUIVALENCE (CSOCC(1),DATA30(543)),(PSOCC(1),DATA30(565)),
     *(SOCC1M(1),DATA30(587)),(SVOL1M(1),DATA30(609)),
     *(CSVOL(1),DATA30(631)),(PSVOL(1),DATA30(653)),
     *(SSPD(1),DATA30(675)),(LNSTS(1),DATA30(697))
C
C
C CONSOLIDATE LANE DATA INTO STATION DATA
C
C
```

```
      DO 190 I=1,NSTNS
      J=NSTNS-I+1 ;STN NO
      PSOCC(J)=CSOCC(J)
      PSVOL(J)=CSVOL(J)
      I1=3*I-2 ;MOST STNS HAVE 3 LANES
      I2=I1+2
      IF(J.LT.6)I2=I2+1 ;STN6 HAS 4 LANES
      IF(J.LT.6)I1=I1+1
      NPOS=0
      VOLSUM=0
      OCCSUM=0
      DO 150 IK=I1,I2
      IF(DNOCC(IK).LT.0)GOTO 150 ;IF OCC IS WRONG,SO IS VOL
      NPOS=NPOS+1
      VOLSUM=VOLSUM+DNVOL(IK) ;ADD LANE VOL
      OCCSUM=OCCSUM+DNOCC(IK) ;ADD LANE OCC
150   CONTINUE
      IF(NPOS.EQ.0)GOTO 180 ;IF OCC WRONG ,SO ARE VOL,SPD,V. TYPES
      NLANE=3
      CSOCC(J)=OCCSUM*NLANE/NPOS ;LATEST 30 SEC STN OCC
      CSVOL(J)=VOLSUM*NLANE/NPOS ;LATEST 30-SEC STN VOL
      SOCC1M(J)=CSOCC(J)+PSOCC(J) ;LATEST 1-MIN OCC
      SVOL1M(J)=CSVOL(J)+PSVOL(J)
      IF(PSOCC(J).LT.0)SOCC1M(J)=-1 ;IF PAST 30-SEC DATA IS WRONG
      IF(PSVOL(J).LT.0)SVOL1M(J)=-1 ;THEN 1-MIN DATA IS ALSO WRONG
      SPDSUM=0
      NSP=0
      NPOS=0
      DO 160 IK=I1,I2
      IF(LNSPD(IK).LT.0)GOTO 160 ;IF SPD IS WRONG,SO IS V. TYPE
      NPOS=NPOS+1
      IF(LNSPD(IK).GT.0)NSP=NSP+1 ;NO OF POS SPEEDS
      SPDSUM=SPDSUM+LNSPD(IK) ;ADD LANE SPEEDS
160   CONTINUE
      IF(NPOS.EQ.0)GOTO 140 ;ALL LANE DATA NO GOOD
      SSPD(J)=SPDSUM
      IF(NSP.GT.0)SSPD(J)=SPDSUM/NSP ;AVG STN SPEED
      GOTO 110
180   CSOCC(J)=-1
      SOCC1M(J)=-1
      SVOL1M(J)=-1
      CSVOL(J)=-1
140   SSPD(J)=-1
110   LNSTG(J)=NPOS ;NO OF 'GOOD' LANES AT THIS STN
190   CONTINUE
C
C
C COMPUTE SECTION TRAVEL TIMES
C
      NSTNM2=NSTNS-2
      DO 500 I=1,NSTNM2
      I1=I ;UPSTR STN
      IF(I.GE.3)I1=I+1
      I2=I+1 ;DOWNSTR STN
      IF(I.EQ.2)I2=I+2
      ISPD=(SSPD(I1)+SSPD(I2))/2 ;AVG SECTION SPEED
      [illegible]
      IF(SSPD(I1).LT.0.AND.SSPD(I2).LT.0)TT(I)=-1
      IF(SSPD(I1).LT.0)ISPD=SSPD(I2)
      IF(SSPD(I2).LT.0)ISPD=SSPD(I1)
      IF(ISPD.GT.0)TT(I)=DIST(I)/ISPD
500   CONTINUE
C
      DO 600 I=1,3
600   DATA30(I)=[illegible]
      DATA30(4)=CONTRL ;CONTROL PERIOD
      DO 602 I=1,NRAMPS
      J=(I-1)*4+5
      DATA30(J)=NODCYC(I)
      DATA30(J+1)=RAMCYC(I)
      DATA30(J+2)=QWAIT(I)
      DATA30(J+3)=QLNTH(I)
602   CONTINUE
      CALL OPEN(3,"DATA30",2,IER,2*64)
      CALL WRITR(3,NL30S,DATA30,12,IER)
      NL30S=NL30S+12
      CALL CLOSE(3,IER)
C
C ACCUMULATE 5-MIN DATA
C
      CALL OPEN(3,"DATA5M",2,IER,2*64)
      CALL READR(3,K5M,DATA5M,7,IER)
      DO 612 I=1,3
612   DATA5M(I)=DATA30(I)
      DO 620 I=4,430
      J=I+43
      IF(I.GE.360)J=I+112 !BYPASS BPD DATA
      IF(COUNT.EQ.1)GOTO 610
      IF(DATA30(J).LT.0.OR.DATA5M(I).LT.0)DATA5M(I)=-1
      IF(DATA5M(I).GE.0)DATA5M(I)=DATA5M(I)+DATA30(J) !ACC 5-MIN DATA
      GOTO 620
610   DATA5M(I)=DAJA30(J) !FIRST 30-SEC DATA
620   CONTINUE
      CALL WRITR(3,K5M,DATA5M,7,IER)
      CALL CLOSE(3,IER)
      COUNT=COUNT+1
      IF (.NOT. ( (MOD(HMS(2),5).EQ.0) .AND. (HMS(3).LT.30) ) )
     *        GO TO 900
      COUNT=1 !5-MIN ALREADY
      IMINS=1 !5 MINS ALREADY
      K5M=K5M+7 !REC FOR NEXT 5 MINS
      GO TO 900
C
900   CONTINUE
      RETURN
      END
      COMPILER NOSTACK
      OVERLAY OVTLU
      SUBROUTINE VTLU
      COMMON/RREC/KL30S,NNREC,KINC,KSM,NALREC,KDISP
      COMMON/NNDAY/NNS(3),NDT(3)
      COMMON/INCA/INCTV(84),INCREC(21),INCFLAG(21),INCSTAT(21)
C
      COMMON/INCB/INCPRG(21),INCOT,INCVT,INCST
      COMMON/RSFIL/RSRCD,RSDVR(21)
C
      INTEGER NNS,NDT,INCREC,INCSTAT
      INTEGER INCPRG,INCUT,INCVT,INCST,RSRCD,RSDVR
      INTEGER LNOCC(66),LNVOL(66),LNSPD(66),STNOCC(22),STNVOL(22),STNSPD(22)
      INTEGER DRIV,CNTR,PASS
      INTEGER OCCVOT(3),VOLVOT(3),SPDVOT(3),OCCMAJ,VOLMAJ,SPDMAJ
      INTEGER INCLOG
      INTEGER INCLOG(25),RESDV(26)
      INTEGER STN,I
      INTEGER DNOCC,DNVOL,DNSPD
      INTEGER OCNT,VLNT,SPNT
C
      DATA OCNT/1/, SPNT/1/, VLNT/1/
C
C UTILIZE INCSTAT ARRAY TO DETERMINE IF/AND WHERE AN INCIDENT OCCURRED
C
      DO 20 STN=1,21
         IF(STN.EQ.3)GOTO 20;                           STATION THREE HAS NO DOWNSTREAM STATION
C
C DETERMINE WHETHER OR NOT A NEW INCIDENT HAS BEEN DETECTED
C
            IF(INCSTAT(STN).EQ.INCPRG(STN))GOTO 20;     NO NEW INCIDENT
            INCPRG(STN)=1                               NEW INCIDENT
         CALL OPEN(3,'RESDEV',2,IER,52)
         CALL READR(3,RSDVR(STN),RESDV,1,IER)
         CALL CLOSE(3,IER)
         DO 31 I=1,3
           OCCVOT(I)=RESDV(14+I)
           VOLVOT(I)=RESDV(18+I)
           SPDVOT(I)=RESDV(22+I)
   31      CONTINUE
C
C
C IMPLEMENT SECOND LEVEL DECISION ROUTINE - PRIMARY VTLU(VOTE TAKING LOGIC UNIT)
C USE DATA COMPILED BY CLDU'S
C TALLY OCCVOT DATA
C
C
         OCCMAJ=0;                                      IF NOT CHANGED(REMAINS ZERO) A NULL (TIE)VOTE HAS RESULTED
         IF(OCCVOT(1).EQ.OCCVOT(2))OCCMAJ=OCCVOT(1);    IF TWO VOTES ARE
         IF(OCCVOT(1).EQ.OCCVOT(3))OCCMAJ=OCCVOT(1);    THE SAME ,THEN THEY FORM A
         IF(OCCVOT(2).EQ.OCCVOT(3))OCCMAJ=OCCVOT(2);    TWO OUT OF THREE MAJORITY
C
C TEST VOL. RESULTS NEXT
C
         VOLMAJ=0
         IF(VOLVOT(1).EQ.VOLVOT(2))VOLMAJ=VOLVOT(1)
         IF(VOLVOT(1).EQ.VOLVOT(3))VOLMAJ=VOLVOT(1)
         IF(VOLVOT(2).EQ.VOLVOT(3))VOLMAJ=VOLVOT(2)
C
C TEST SPD. RESULTS
C
         SPDMAJ=0
         IF(SPDVOT(1).EQ.SPDVOT(2))SPDMAJ=SPDVOT(1)
         IF(SPDVOT(1).EQ.SPDVOT(3))SPDMAJ=SPDVOT(1)
         IF(SPDVOT(2).EQ.SPDVOT(3))SPDMAJ=SPDVOT(2)
C
C
C AFTER COMPLETING PRIMARY VTLU - DETERMINE FINAL DECISION USING SECONDARY VTLU
C WITH WEIGHTINGS - BY MAJORITY RULE
C SET COUNTERS TO ZERO
C
C
         DRIV=0
         CNTR=0
         PASS=0
C
C DETERMINE VOTE CONTAINED IN OCCMAJ AND INCREMENT APPROPRIATE COUNTER BY OCNT (TO ACCOUNT FOR WEIGHTING)
C
         IF(OCCMAJ.EQ.1)DRIV=DRIV+OCNT
         IF(OCCMAJ.EQ.2)CNTR=CNTR+OCNT
         IF(OCCMAJ.EQ.3)PASS=PASS+OCNT
C
C REPEAT WITH VOL. AND SPEED
C
```

```
      IF(VOLMAJ.EQ.1)DRIV=DRIV+VLNT
      IF(VOLMAJ.EQ.2)CNTR=CNTR+VLNT
      IF(VOLMAJ.EQ.3)PASS=PASS+VLNT
C
      IF(SPDMAJ.EQ.1)DRIV=DRIV+SPNT
      IF(SPDMAJ.EQ.2)CNTR=CNTR+SPNT
      IF(SPDMAJ.EQ.3)PASS=PASS+SPNT
C
C COMPARE TOTALS - SET RESULT IN INCLAN TO LANE WITH MOST VOTES
C
      IF(DRIV.GT.CNTR)GOTO 70
        IF(CNTR.GT.PASS)GOTO 72;                          CENTER LANE INCIDENT
 70       IF(DRIV.GT.PASS)GOTO 71;                        DRIVING LANE INCIDENT
            INCLAN=3;                                     PASSING LANE INCIDENT
            GO TO 75
C
 71         INCLAN=1;  INC. IN DRIVING LANE
          GO TO 75
C
 72        INCLAN=2;     INC. IN CENTER LANE
C
C
C ONCE A DECISION IS REACHED, WRITE LANE OF INCIDENT INTO RECORD OF "INCLOG"
C                             FOR THAT INCIDENT
C
 75      CALL OPEN(3,'INCLOG',2,IER,20)
         CALL READR(3,INCREC(STN),INCLOG,1,IER)
         INCLOG(5)=INCLAN
         CALL WRITR(3,INCREC(STN),INCLOG,1,IER)
         CALL CLOSE(3,IER)
C
C
C CREATE ARRAY 'REDOV' AND WRITE INTO DISK FILE
C
C
         REDOV(1)=MDY(3)
         REDOV(2)=MDY(1)
         REDOV(3)=MDY(2)
         REDOV(4)=HMS(1)
         REDOV(5)=HMS(2)-5
         REDOV(6)=HMS(3)
         IF(REDOV(5).GE.0)GOTO 80
            REDOV(4)=REDOV(4)-1
            REDOV(5)=55
 80      REDOV(7)=STN
         REDOV(8)=INCLAN
         REDOV(9)=INCDT
         REDOV(10)=[illegible]
         REDOV(11)=INCST
         REDOV(12)=OCNT
         REDOV(13)=VLNT
         REDOV(14)=SPNT
         REDOV(18)=OCCMAJ
         REDOV(22)=VOLMAJ
         REDOV(26)=SPDMAJ
C
C
C
         CALL OPEN(3,'REDOV',2,IER,52)
         CALL WRITR(3,REDREC(STN),REDOV,1,IER)
         CALL CLOSE(3,IER)
C
 20      CONTINUE
       RETURN
       END
      [illegible]
      SUBROUTINE MOD030
      DIMENSION DATASM(408)
      DIMENSION SDVOL(46),SDOCC(46)
      DIMENSION DNVOL(67),UPVOL(67),DNOCC(67),UPOCC(67),LNSPD(67)
      DIMENSION LVCNT(67)
      DIMENSION CDOCC(22),PSOCC(22),SOCCIM(22),SVOLIM(22),CSVOL(22),
     *PSVOL(22),SSPD(22),LNSTS(22)
      [illegible]
      COMMON/CTRL1/CONTRL,MAXCYC,METER(10),MODCYC(10)
      COMMON/CTRL2/CYCOVR(10),RAMCYC(10)
      COMMON/WKDAT/WKD(3),HDY(3)
      COMMON/SLOW/STOT,DTOT,NRAMPS,NSTNS,CAP(22)
      COMMON/DECTS/DIST(20)
      COMMON/T18/TT(20)
      COMMON/D30/DATA30(710)
      [illegible]
      COMMON/TCNT/COUNT,SCNTSM,IMINS,XCOUNT
      COMMON/MREC/SL30B,NNREC,KINC,ESM,MALREC,KDISP
C
C
C  THE VARIABLE 'BREC' IS  FOR TEST PURPOSES ONLY AND MUST BE REMOVED BEFORE
C  IMPLEMENTING THIS ROUTINE IN THE SYSTEM
C
      COMMON/RCRD/BREC
      INTEGER [illegible]
C
C
      INTEGER DATASM
      INTEGER DWAIT,QLNTH
      INTEGER CONTRL,MAXCYC,METER,MODCYC
      INTEGER CYCOVR,RAMCYC
      INTEGER SDVOL,SDOCC
      INTEGER DNVOL,UPVOL,DNOCC,UPOCC,LNSPD,LVCNT
      INTEGER CDOCC,PSOCC,SOCCIM,SVOLIM,CSVOL,PSVOL,SSPD,LNSTS
      INTEGER WKD,HDY
      INTEGER STOT,DTOT,NRAMPS,NSTNS,CAP
      INTEGER DIST
      INTEGER TT
      INTEGER DATA30
      INTEGER XREFTB,XREF
      INTEGER COUNT,SCNTSM,IMINS,XCOUNT
      INTEGER SL30B,NNREC,KINC,ESM,MALREC,KDISP
      INTEGER SSTAT,DNSTAT,UPSTAT
      INTEGER SPDS,VOLSUM,OCCSUM,SPDSUM,NSP
      INTEGER I,K,J,I1,I2,IK,NLANE,NSTNP,ISPD
      EQUIVALENCE (SDVOL(1),DATA30(49)),(SDOCC(1),DATA30(95)),
     *(DNVOL(1),DATA30(141)),(UPVOL(1),DATA30(208)),
     *(DNOCC(1),DATA30(275)),(UPOCC(1),DATA30(342)),
     *[illegible]
      EQUIVALENCE (CDOCC(1),DATA30(543)),(PSOCC(1),DATA30(565)),
     *(SOCCIM(1),DATA30(587)),(SVOLIM(1),DATA30(609)),
     *(CSVOL(1),DATA30(631)),(PSVOL(1),DATA30(653)),
     *(SSPD(1),DATA30(675)),(LNSTS(1),DATA30(697))
C
C  CHECK FOR MALFUNCTION IN SINGLE-LOOP DET DATA
C
      DO 200 I=1,STOT
      SSTAT=XREFTB(I)/1000 !GET STATUS
      IF(SSTAT.EQ.0)GOTO 200 !DET OK
      SDVOL(I)=-1 !SET VOL=-1
      SDOCC(I)=-1 !SET OCC=-1
200   CONTINUE
C
C  CHECK FOR MALFUNCTION IN DBLE-LOOP DET DATA
C
      DO 300 I=1,DTOT
      K=I+STOT
      DNSTAT=XREFTB(K)/1000 !GET STATUS
      UPSTAT=XREFTB(K+DTOT)/1000
      IF(DNSTAT.EQ.0.AND.UPSTAT.EQ.0)GOTO 300 !BOTH OK
      IF(DNSTAT.EQ.UPSTAT)GOTO 340 !BOTH NOT OK
      IF(DNSTAT.EQ.0)GOTO 350 !DOWN OK BUT NOT UP.
      DNVOL(I)=UPVOL(I) !USE UPSTR.VOL - UP OK BUT NOT DOWN
      DNOCC(I)=UPOCC(I) !USE UPSTR. OCC
      GOTO 320
340   DNVOL(I)=-1  !SET DOWN VOL & OCC=-1
      DNOCC(I)=-1
350   UPVOL(I)=-1
      UPOCC(I)=-1  !SET UP VOL & OCC=-1
320   LNSPD(I)=-1  !SET LANE SPEED =-1
      LVCNT(I)=-1  !SET LONG VEH COUNT=-1
300   CONTINUE
C
C  TEMPORARY ROUTINE TO OBTAIN USABLE DATA IN ABSCENSE OF COLLECTING STATIONS
C
```

```
      [illegible]
      CALL RDRW(3,NREC,LOGRD30,12,IER,NBYTF)
      IF(IER.NE.1)GOTO 51
      DO 50 IELMT=1,675
50       DATA30(IELMT)=LOGRD30(IELMT)
      NREC=NREC+12
      CALL CLOSE(3,IER)
      RETURN
51    IF(IER.EQ.9)TYPE "END OF FILE <DISK30>"
      IF(IER.NE.9)TYPE "ERROR CODE: ",IER," RETURNED BY LIBRARY SUBROUTINE  RDRW"
      CALL CLOSE(3,IER)
      RETURN
      END
```

```
;       [illegible] - FORTRAN IV MULTI-TASKING [illegible]
        .TITL CLOCK
; INTERRUPT SERVICE FOR USER CLOCK. IT ACTIVATES
; A TASK WHICH IS WAITING FOR A MESSAGE TO
; BE DEPOSITED IN A 1-WORD INTEGER VARIABLE
; CALLED "TIMER", IN LABELLED COMMON BLOCK "KLOCK".
;
        .ENT    CLOCK
        .EXTN   .IXMT,.UCEX
        .COMM   KLOCK,1
        .NREL
;
; REFER TO RDOS REFERENCE MANUAL, PG. 5-11, FOR .IXMT,
;    AND TO USER DEVICE DRIVER IMPLEMENTATION IN RDOS MANUAL, PG. 3-9, FOR .UCEX'
;
CLOCK:  STA     3,TEMP3         ;SAVE AC3
        STA     2,TEMP2         ;SAVE AC2
        LDA     0,.MESS         ;YES,TIME TO CALL TASK
        SUBZR   1,1
        .IXMT                   ;WAKE UP TASK
        JMP     .+1             ;ERROR?
        SUBZR   1,1
        LDA     3,TEMP3         ;RESTORE AC3
        LDA     2,TEMP2         ;RESTORE AC2
        .UCEX                   ;EXIT
;
.MESS:  .GADD   KLOCK,0
TEMP2:  0
TEMP3:  0
;
        .END
```

```
;  THIS HAS BEEN MODIFIED ON 81-07-28 BY [illegible]:
;  THE REQUIRED MODIFICATIONS WERE DESIGNED BY DAVE DUNLOP.
;
;  THIS IS AN ASSEMBLER LANGUAGE PACKAGE WHICH ALLOWS THE NOVA 4/X
;  FORTRAN IV PROGRAMMER TO COMMUNICATE WITH THE ISI/SP MODEL 300
;  NETWORK CONTROLLER IN AN RDOS MULTI-TASKING ENVIRONMENT.
;
        .TITL   ISDEF
        .ENT    ISDEF,BREAD,BWAIT
        .EXTN   .UIEX
        .EXTU
;
        .ZREL
.RG1:   ARCDE
.WR1:   WRCDE
        .NREL
;
;  DEVICE CONTROL TABLE (DCT)
;  ==========================
;
        .DUSR   ISI=15                  ;ISI/300 DEVICE CODE DEF'N
ISDCT:  ISS+.+1                         ;ADDRESS OF ISI DCT
        ISSA                            ;INTERRUPT STATE SAVE AREA
        077                             ;MASK OUT ALL SLOWER DEV & DISK
        ISINT                           ;INTERRUPT HANDLER
;
;  ADDITIONAL STORAGE
;  ==================
;
ISSA:   .BLK    10                      ;STATE SAVE AREA
ISDCD:  ISI                             ;ISI/300 DEVICE CODE
ISMSG:  0                               ;MESSAGE ADDRESS
ISTAT:  .BLK    1                       ;I/O STATUS WORD
ARCDE:  0
WRCDE:  0
;
;  INTERRUPT SERVICE ROUTINE
;  =========================
;
ISINT:  NIOC    ISI                     ;SIGNIFY INTERRUPT
        STA     0,TMP1                  ;SAVE AC'S
        STA     3,TMP2
        SUBZL   1,1                     ;SET UP MESSAGE
        STA     1,ISMSG                 ;SET MESSAGE LOCATION
        SUBZL   1,1                     ;[illegible] RE-SCHEDULING
        LDA     0,TMP1                  ;RESTORE AC'S
        LDA     3,TMP2
        .UIEX
TMP1:   0
TMP2:   0
;
;  "ISDEF" IS CALLED ONLY ONCE, AT THE BEGINNING OF THE PROGRAM.
;  IS PERFORMS INITIALIZATION.  IT ATTACHES THE DEVICE TO THE
;  SYSTEM AND SETS UP THE INTERRUPT HANDLER.
;
;  CALLED BY
;  =========
;                CALL ISDEF(IER)
;  WHERE:
;       IER = ERROR CODE RETURNED TO CALLER
;               (FAR CONVENTION - IE. 1=OK)
;
.IER=   -167
ISDEF:  JSR     @.CPYL
        SUB     0,0
        STA     0,ISTAT                 ;INITIALIZE STATUS
        LDA     0,ISDCD                 ;ISI DEVICE CODE
        LDA     1,ISDCT                 ;DCT ADDRESS
        LDA     2,C3                    ;# OF 1K BLOCKS FOR MAP
        .SYSTM
        .IDEF                           ;INTERRUPT DEFINE
        JMP     ISD20                   ;ERROR RETURN
        LDA     2,D2                    ;OK RETURN CODE
ISD20:  INC     2,2                     ;CONVERT RDOS CODE
        INC     2,2                     ;   INTO
        INC     2,2                     ;FORTRAN ERROR CODE
        STA     2,@IER,3                ;STUFF RETURN CODE
        JSR     @.FRET                  ;EXIT
C3:     3
D1:     -1
D2:     -2
;
;  "BREAD" IS CALLED FROM A FORTRAN TASK TO READ THE ISI 300. THE
; TASK IS SUSPENDED UNTIL THE I/O IS COMPLETE.
;
; CALLED BY
; ---------
;               CALL BREAD(IBUFF(1),LGTH,IER)
;
; WHERE:
;        IBUFF = INPUT BUFFER ADDRESS
;        LGTH  = [illegible] OF WORDS TO BE READ.
;        IER   = STATUS CODE RETURNED TO CALLER
;
IBUFF=  -167
LGTH=   -166
IER=    -165
[illegible]=  -164
VALUE=  -163
        ;
BREAD:  JSR     @.CPYL
        LDA     0,ISTAT         ;GET DEVICE STATUS
        MOV     0,0,SZR         ;CHECK IT
        JMP     [illegible]     ;ERROR - I/O IN PROGRESS
        LDA     0,[illegible]   ;SET INPUT STATUS
[illegible]: STA 0,ISTAT        ;SAVE STATUS
        LDA     0,[illegible]
        LDA     1,IBUFF,3       ;GET USER BUFFER ADDRESS
        [illegible]
        [illegible]
        JMP     [illegible]
        MOV     1,0
        [illegible] 0,[illegible] ;LOAD DEVICE ADDRESS CTR
        LDA     0,@LGTH,3       ;GET WORD COUNT
        NEG     0,0             ;2'S COMPLEMENT
        [illegible] 0,[illegible] ;LOAD DEVICE WORD COUNTER
        [illegible]             ;[illegible]
        STA     0,[illegible]
        LDA     0,ISTAT
        [illegible] 0,[illegible] ;START I/O
LP:     LDA     0,[illegible]
        MOV     0,0,[illegible]
        JMP     LP
        LDA     2,[illegible]   ;I/O COMPLETE
        [illegible]
[illegible]: LDA 0,[illegible]  ;GET ERROR CODE
[illegible]: [illegible]        ;CONVERT [illegible] CODE
        [illegible]             ;  INTO
        [illegible]             ;FORTRAN ERROR CODE
        STA     0,@IER,3        ;STORE RETURN CODE
        [illegible]             ;RESET STATUS
        STA     0,ISTAT
        [illegible]             ;RETURN
SAVE:   0
;
; "BWRIT" IS CALLED FROM A FORTRAN TASK TO WRITE TO THE [illegible].
; THE TASK IS SUSPENDED UNTIL THE I/O IS COMPLETE.
;
; CALLED BY
; ---------
;               CALL BWRIT(IBUFF(1),LGTH,IER)
; WHERE:
;        IBUFF = OUTPUT BUFFER ADDRESS
;        LGTH  = NUMBER OF WORDS TO BE WRITTEN
;        IER   = STATUS CODE RETURNED TO CALLER
;
        ;
BWRIT:  JSR     @.CPYL
        LDA     0,ISTAT         ;CHECK DEVICE STATUS
        MOV     0,0,SZR
        JMP     [illegible]     ;ERROR - GO RETURN
        LDA     0,[illegible]   ;SET OUTPUT STATUS
        JMP     [illegible]     ;GO PROCESS
;
        .END
```